Projection-based model order reduction for aerodynamic applications

Von der Carl-Friedrich Gauß Fakultät
Technische Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigte Dissertation

von

Alexander Vendl

geboren am 04.01.1984

in Gifhorn

Eingereicht am:	21.01.2013
Mündliche Prüfung am:	31.05.2013
Referentin:	Prof. Dr. Heike Faßbender
Korreferent:	Prof. Dr. Nicolas Gauger

Technische Universität Braunschweig

(2013)

Vendl, Alexander:
Projection-based model order reduction
for aerodynamic applications,
Dissertation, Technische Universität
Carolo Wilhelmina, Braunschweig, 2013.

Kurzfassung

Die vorliegende Arbeit hat die Modellreduktion von Erhaltungsgleichungen der Strömungsmechanik zum Thema. Hierbei steht die Entwicklung eines effizienten, auf Projektion basierenden Modellreduktionsverfahren zur Lösung dieser nichtlinearen Gleichungen im Fokus.

Die europäische Flugzeugindustrie steht vor grossen Herausforderungen. Aufgrund des Klimawandels und zur Steigerung der Wirtschaftlichkeit ihrer Produkte hat sie sich zum Ziel gesetzt, die Emissionen ihrer Flugzeuge drastisch zu senken. Dies ist zum Beispiel durch neuartige Flugzeugformen zu bewerkstelligen. Hierbei wird der numerischen Simulation der Strömungseigenschaften des Flugzeugs einen grossen Stellenwert eingeräumt.

Obwohl der Einsatz der numerischen Strömungssimulation gegenüber experimentellen Tests im Windkanal an Bedeutung gewinnt, schränkt der hohe zeitliche Aufwand für die große Anzahl der benötigten Berechnungen die Anwendbarkeit ein. Daher spielen effiziente Berechnungsmethoden wie die Modellreduktion in diesem Kontext eine wichtige Rolle. Die Modellreduktion hat zum Ziel, die Anzahl der Gleichungen des zugrundeliegenden Systems beispielsweise durch Projektion auf einen niedrig-dimensionalen Unterraum zu reduzieren. Das Modell reduzierter Ordnung sollte dann die Eigenschaft haben, dass es sich deutlich effizienter lösen lässt.

Während Modellreduktion schon sehr erfolgreich auf lineare Probleme angewendet wurde, sind in jüngster Vergangenheit Verfahren entwickelt wurden, die nichtlineare Problemstellungen effizient lösen können. Die Herausforderung bei der Anwendung auf nichtlineare Gleichungen wie etwa der Erhaltungsgleichungen der Strömungsmechanik besteht darin, dass obwohl die Anzahl der diskretisierten Gleichungen sehr einfach durch Projektion reduziert werden können, keine Unabhängigkeit von der Ordnung des Ausgangsproblems erreicht wird. Dies liegt darin begründet, dass in jedem iterativen Schritt zur Lösung des reduzierten Modells die nichtlineare rechte Seite des ursprünglichen Systems ausgewertet wird, die von der Größenordnung des Ausgangsproblems ist. Daher ist das Ziel von nichtlinearen Modellreduktionsverfahren, die reduzierten Modelle so zu erstellen, dass die rechte Seite der Erhaltungsgleichungen nicht an jedem einzelnen Punkt des Rechengitters, sondern an einer Teilmenge dieser Punkte ausgewertet wird.

In dieser Arbeit wird eine Methode namens „Missing Point Estimation“ verwendet. Diese erreicht das obige Ziel mit Hilfe einer geeigneten Projektion. Mit anderen Worten ist die Projektion gerade so konstruiert, dass die rechte Seite nur an den ausgewählten Punkten berechnet werden muss. Zudem ist aufgrund dessen, dass auf einen niedrig-dimensionalen Unterraum projiziert wird, die Anzahl der Gleichungen stark gegenüber derjenigen des Ausgangsproblems reduziert. Dies hat dann insgesamt zur Folge, dass sich das reduzierte Modell effizient lösen lässt.

Um einen niedrig-dimensionalen Unterraum zu bestimmen, wird die „Proper Orthogonal Decomposition“ verwendet. Diese Methode bedient sich sogenannter „Snapshots“, welche für das zugrundeliegende System charakteristische Lösungen sind, die zunächst berechnet werden müssen oder bereits vorliegen. Durch Speicherung dieser Lösungen in einer Matrix und anschließender Berechnung der Singulärwertzerlegung dieser Matrix erhält man eine Basis, welche aus den linken Singulärvektoren besteht. Diese Basis wird dann in der Missing Point Estimation zum einen für die Darstellung der Lösung und zum anderen für die Projektion auf den durch ihr aufgespannten Unterraum verwendet.

Bei der Verwendung der Missing Point Estimation für unterschiedliche Anwendungsgebiete ist offensichtlich die Auswahl der Punkte, an denen die rechte Seite der Erhaltungsgleichungen evaluiert werden, sehr verschieden. In dieser Arbeit soll untersucht werden, welche Punktauswahlen sich besonders für die Vorhersage von Strömungsfeldern um Tragflächenprofile als auch um komplexe drei-dimensionale Flugzeugkonfigurationen eignen. Hierbei steht vor allem die Anwendung auf industrierelevante Testbeispiele im Vordergrund.

Die Ergebnisse dieser Arbeit sind im Rahmen des Verbundprojekts „ComFliTe“ (Computational Flight Testing) entstanden. Dieses Projekt wurde durch das Bundesministerium für Wirtschaft und Technologie gefördert und hatte den Ausbau der Fähigkeiten der numerischen Simulation im Bereich der Aerodynamik zum Ziel. Insbesondere wurde in einem Arbeitspaket Modellreduktionsverfahren in der Strömungssimulation untersucht.

Abstract

The subject of this thesis is model order reduction for the governing equations arising in computational fluid dynamics. It focuses on the development of an efficient, projection-based reduced order model for solving these nonlinear equations.

The European aviation industry faces great challenges. Due to the climate change and to increase the profitability of their products, it has set itself the goal to drastically reduce the emissions of their aircrafts. This can be accomplished with the help of innovative airplane shapes for example. Hence, the numerical simulation of the aircraft's flow properties plays an important role.

Although the application of numerical simulations gains in importance compared to experimental wind tunnel tests, the large amount of time needed for the vast number of computations limits the applicability. As a result efficient computational methods, such as model order reduction, play an important role. The goal of model order reduction is to reduce the number of equations of the underlying system, for example, with the help of projecting the governing equations onto a low-dimensional subspace. The reduced order model should have the property that it can be solved much more efficiently.

While model order reduction has been successfully applied to linear problems, more recently techniques have been developed for tackling nonlinear problems as well. The challenge of the application to nonlinear equations like the governing equations of CFD is that, although the number of discretized equations can easily be reduced by projection, an independence from the full order is not actually achieved. This is due to the fact that in each iterative step of solving the reduced model, a nonlinear right hand side has to be evaluated, which is of the order of the original order. As a result nonlinear model order reduction methods aim at creating reduced order models, which do not evaluate the right hand side of the governing equations at each and every computational grid point, but only at a small subset of these points.

In this work a method called missing point estimation is used. It achieves the above goal with an appropriate projection. In other words, the projection is constructed such that the right hand side of the governing equations is evaluated only at selected points. Furthermore, due to the projection onto a low-dimensional subspace, the number of equations is significantly reduced compared to the original problem. This altogether yields a reduced order model, which can be solved efficiently.

In order to construct a low-dimensional subspace proper orthogonal decomposition is used.

This method employs so-called snapshots, which are solutions to the governing equations characteristic of the model to be constructed. These snapshots either have to be computed or are already available. By storing them in a matrix and computing the singular value decomposition of this matrix, a basis is obtained, which consists of the left singular vectors. This basis is utilized in the method of missing point estimation for representing the solutions as well as for the projection onto the subspace spanned by it.

When applying missing point estimation to different fields of application, the selection of the points differs considerably. In this work it shall be investigated, which point selections are most suitable for the prediction of the flow fields around airfoils and complex three-dimensional aircraft configurations. Special attention is given to industrially relevant test cases.

The results of this thesis have been obtained in the research project ComFliTe (Computational Flight Testing). This project has been funded by the German "Federal Ministry of Economics and Technology" (BMWi) and had as an overall goal to advance the capabilities of numerical simulations in the field of aerodynamics. In particular, model order reduction techniques for computational fluid dynamics have been investigated in one of the work packages.

Acknowledgement

First of all, I would like to thank my advisor, Heike Faßbender, for giving me the opportunity to be her PhD student and for all the valuable advice she has given to me. Also, I would like to acknowledge the many national and international conferences that she has made possible in my PhD time. Most importantly, I appreciate the opportunity to give a talk at the SIAM Annual Meeting 2010 as well as at the ICIAM in 2011 and exchange ideas with experts from the field of model order reduction.

Furthermore, I would like to express my heartfelt gratitude to the collaborators in the ComFliTe project from the German Aerospace Center (DLR) in Braunschweig – Stefan Görtz, Ralf Zimmermann, Michael Mifsud and Jan Sippli. Without all their help accomplishing this work would have been that much harder. I would like to thank Stefan for the great organization of the project and his valuable ideas and comments about my work; Ralf for the numerous discussions, hints, source code, advice and support; Michael for all the snapshot computations that he has done for me and all his advice from an application's point of view; and Jan for efficiently glueing the code fragments together with python and all his computer science advice.

Another group, which deserves my credit, are my colleagues at the TU Braunschweig. I would like to give thanks to Matthias Bollhöfer for his help with integrating LAPACK into my C code and for answering all my questions; to André Bodendiek for listening to my babbling about projections and for always having the door open to ask for quick help; Andreas Soppa for reading my papers, for listening and his career advice; Juan Duran Amorocho for his help with LINUX as well as MAKE and the discussions; Peter Stange and André Eppler for their support and our lunchtime discussions.

Besides the numerics group, I would like to thank Rainer Hempel for his inspiring lectures and Martina Pawlak, Marko Stautz and Philipp Öffner for answering my theoretical math questions.

Apart from the technical support I would like to thank my parents and my brother most sincerely for giving me the strength and endurance to finish my PhD. Without their assistance I most certainly would have had a much harder time during my PhD. The same holds true for all my friends, most importantly Holger Müller who always psyched me up when times were rough and who always had time to talk.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

CFD	Computational Fluid Dynamcis
MOR	Model Order Reduction
MPE	Missing Point Estimation
ODE	Ordinary Differential Equation
POD	Proper Orthogonal Decomposition
ROM	Reduced Order Model
SVD	Singular Value Decomposition

Physical Variables

$\dot{Q}$	heat transfer
$\dot{W}$	power
γ	ratio of specific heats
m	mass
μ	coefficient of shear viscosity
Ω	control volume
Ω_{CM}	volume occupied by control mass
ρ	density
$\overline{\overline{\tau}}$	stress tensor
$\overline{\overline{F}}$	flux tensor
$\overline{\overline{F}}_c$	convective flux tensor
$\overline{\overline{F}}_v$	viscous flux tensor
$\boldsymbol{F}$	forces
$\boldsymbol{F}_b$	integrated body forces
$\boldsymbol{f}_b$	body forces
$\boldsymbol{F}_S$	integrated surface forces
$\boldsymbol{I}$	momentum
$\boldsymbol{K}$	heat transfer vector
$\boldsymbol{Q}$	source term
$\boldsymbol{v}$	Cartesian velocities
$\boldsymbol{W}$	continuous vector of conservative variables
$\boldsymbol{w}$	discrete vector of conservative variables
$\boldsymbol{F_x}$	flux vector
$\boldsymbol{F_x}_c$	convective flux vector
$\boldsymbol{F_x}_v$	viscous flux vector
c_p	specific heat at constant pressure
c_v	specific heat at constant volume
E	mass-specific total energy
e	inner energy
E_T	total energy
H	mass-specific total enthalpy
k	heat transfer coefficient
l	characteristic size scale of a flow
M	mass matrix
M_∞	free-stream Mach number
p	pressure
R	gas constant
Re	Reynolds number
S	surface of control volume
T	temperature
$\dot{W}_\mathrm{m}$	power of body forces
$\dot{W}_p$	power of pressure
$\dot{W}_v$	power of viscous forces

Variables of Model Order Reduction

V	matrix of right singular vectors
Σ	diaginal matrix of singular values
U	matrix of left singular vectors/change of basis
U_d	untruncated part of basis matrix U
$\bar{U}$	truncated part of the basis matrix U
$\boldsymbol{u}$	POD basis vector
$\boldsymbol{a}$	POD basis coefficient vector
U^{-1}	inverse of change of basis U
W_d^T	untruncated part of basis matrix U^{-1}
$\bar{W}^T$	truncated part of basis matrix U^{-1}
N	number of points of grid
n	number of points selected for MPE
m	number of snapshots
d	number of modes
P	selection matrix

Chapter 1

Introduction

With a vision for 2050, the report of the High Level Group on Aviation Research [15] has set ambitious goals for the European aviation industry. Among others these include a 75% reduction in carbon dioxide (CO_2) emissions per passenger kilometer and a 90% reduction in nitrogen oxides (NO_x). The report also stresses the importance of innovations for ensuring the competitiveness of the European aviation industry.

In order to overcome both of these challenges, numerical simulations will play a crucial role. This is due to the fact that available computing power is increasing exponentially according to Moore's law [43]. For this reason numerical simulations are expected to be more cost efficient in the future [52, Fig. 27]. As a matter of fact, it is envisioned by experts that *digital flight-tests* [32, 52, 55], that is, supporting wind tunnel experiments and flight testing by numerically simulations, are possible in the future. To this end, however, great efforts have to be made.

Nowadays, numerical simulations are used on a daily basis by the aviation industry for the design and development of their aircrafts and help the engineers to analyze the flow behavior of the airplanes in various disciplines [32, 61]. Due to this, wind tunnel testing has decreased considerably [55, 32].

However, to this day, the monetary price for wind tunnel testing remains lower than that of numerical simulations, if a wide range of test cases is considered. This is because, although the initial cost for designing and creating the model for experimental tests is rather high, the cost for the production of data for a single flow condition is very moderate. Thus, the cost increases only slightly with the number of flow conditions tested. On the other hand, for numerical simulations the initial cost, e.g. for grid generation, is comparatively low, while the cost is much higher for a single parameter setting. Therefore, with an increasing number of cases, the overall cost of numerical simulations quickly exceeds that of the experimental tests [52].

A means to reduce the computational cost per simulation and thus increasing the competitiveness of numerical simulations is model order reduction. It aims at creating a reduced order model (ROM), which can be solved in a few seconds or minutes rather than hours, while capturing the relevant behavior of the aircraft at the same time.

In order to be able to understand the idea of model order reduction in more detail, it is essential to look at the mathematical model, which is the foundation of numerical simulations of airplane aerodynamics. It describes how the air flows around the body and the wing of an aircraft. If viscous effects shall be modeled by the mathematical model, the Navier-Stokes equations are used and otherwise the Euler equations. Both are a set of nonlinear partial differential equations, which can be solved by various numerical methods.

Numerical methods transfer the continuous equations into discrete counterparts, which is called discretization. To this end, the space for which the flow has to be computed is divided into small geometrical elements. These geometrical elements are called grid cells and together they make up the computational grid. To the grid cells a so-called grid point is associated, at which the flow variables such as density or pressure are stored. The spatial derivatives of the governing equations are then approximated by replacing the differentials by finite differences between the flow variables of neighboring grid points. This leads to a large system of nonlinear equations. As a matter of fact, for three-dimensional airplane configurations simulated with the Navier-Stokes equations, more than ten million grid points have to be used.

The number of equations of such a system – where each equation is associated with a particular grid point – is also referred to as the order. Hence, the idea of model order reduction is to reduce the large number of equations to a small set of equations. This is typically done by projecting the original set of equations onto a low-dimensional subspace.

While in model order reduction for linear dynamical systems the reduction of the number of equations already suffices to substantially speed up the simulation time, this is not the case for nonlinear systems like the Navier-Stokes and the Euler equations. The bottleneck for these systems is the evaluation of the nonlinear function describing the spatial discretization of the differential equations, which still has the order of the original model, even after the projection onto a low-dimensional subspace. For this reason sparse evaluations of this nonlinear function have been proposed [5, 13], that is, evaluation of the spatial discretization at only some selected locations.

In this work such a nonlinear model order reduction approach, called missing point estimation (MPE) [5], shall be adapted such that it can be used in context of aerodynamic applications. In particular, a suitable set of locations shall be determined, at which the nonlinear function of the spatial discretization is to be evaluated. The author is not aware of any prior applications of missing point estimation to aerodynamic applications.

1.1 State-of-the-art of model order reduction techniques in aerodynamics

The vast majority of the model order reduction techniques in aerodynamics use the so-called proper orthogonal decomposition [26], which is often abbreviated by POD. POD is a means to obtain a set of basis functions, which describe the dynamics of the model at hand. The basis functions are derived from observations of the model, often called snapshots. In context of numerical simulations, these snapshots are solutions to the governing equations for a parameter setting or at a point in time. The POD basis results from a spectral analysis of the snapshots. A detailed description of the proper orthogonal decomposition is given in Chapter 7.

1.1.1 Projection-based model order reduction techniques

While most model order reduction approaches in fluid dynamics have in common that they use POD, they differ considerably in the way that the reduced order model is set up. In the following some of the existing reduced order models based on projection are described, in order to draw a clear distinction to the method proposed in this work.

Galerkin projection

At first, Galerkin projection is considered. Unlike in other scientific disciplines, in context of fluid dynamics, the reduced order model is referred to as Galerkin projection by some authors [40, 39], if the projection onto the POD subspace is applied to the *continuous* governing equations.[1]

By decomposing the governing equations into a linear as well as into a quadratic and a cubic nonlinear differential operator, the projected system can be formulated with the help of inner products between these operators and the POD basis functions [30, 53, 39]. After expressing the solutions in the governing equations in terms of the POD basis functions, the reduced order system is only dependent on the POD basis and the corresponding coefficients. Note that in the reduced order system the number of unknowns, where the unknowns are in fact the POD basis coefficients, is equal to the dimension of the POD subspace and typically very small compared to the dimension of the original model.

In the discretization of the reduced order model, all scalar products between the POD basis functions and the differential operators are computed numerically. These scalar products are of the order of the grid size (number of grid points), which is typically very large. As a result, the discretization is time-consuming. However, when solving the reduced order model, all computations are of the order of the dimension of the POD subspace. Thus, a system, which is independent from the grid size – which is in fact of the order of the original full order model – is obtained and can be efficiently computed.

The Galerkin projection method is mainly applied to unsteady incompressible as well as compressible flow computations with moderate Reynolds numbers ($Re < 10.000$). Among the many applications are a cylindrically propagating and reflecting pulse [30], separation control of the flow around a high-lift configuration with high-frequency actuation [38], and simulating cavity flow oscillations [53].

Subspace projection

Lucia et al. [39] draw a distinction between applying Galerkin projection to the continuous and to the spatially discretized governing equations. In fact, they call the latter approach *subspace projection*. In the subspace projection method the governing equations are discretized first. This yields the so-called residual, which is then projected onto the POD subspace. The projection is expressed by a number of inner products between the residual itself and each of the POD basis functions.

---

[1] If the projection is applied to the spatially discretized governing equations, then it is called subspace projection by some authors [40, 39].

While the projected reduced order system consists of as many equations as there are POD basis functions, which are typically only a few, computing the residual and the inner products is of the complexity of the grid size. Furthermore, since the residual changes in each time step, the inner products cannot be precomputed, but must be computed when solving the projected (reduced) system. Therefore, there is a dependence of the reduced model on the full order. However, it has to be noted that the time step can be chosen much larger than in the original model [39, 40].

The subspace method can be used both for steady and for unsteady simulations. It has been applied to model the quasi-static movement of strong shock waves in a quasi-one-dimensional nozzle [39] and to simulate the steady state, supersonic flow-field around a blunt body [40] for example.

Comparison between Galerkin projection and subspace projection

The main difference between Galerkin and subspace projection is whether the projection is applied to the continuous or the spatially discretized governing equations. Projecting the continuous equations in Galerkin projection offers the possibility to construct a reduced order system, which is indeed independent from the order of the original model, that is, no computation of the complexity of the number of grid points has to be done. This has the obvious advantage that the reduced order model can be solved very efficiently [9]. However, a lot of effort has to be spent on constructing the reduced order model. In addition, Galerkin projection reduced order models are problem-specific according to [9], that is, if the snapshots change, then the reduced order model has to be constructed once again.

Subspace projection, on the other hand, only requires the residual of the governing equations. Any flow solver which is able to return a residual can be used as a black box. Hence, all capabilities such as different discretization schemes and turbulence models can be exploited by the reduced order model. This makes the subspace projection method very flexible.

Yet, the main disadvantage of this method is that there is a dependence on the full order dimension as outlined in the previous subsection. That is, due to the fact that the residual has to be evaluated at each grid point, the complexity of an iteration is of the order of the number of computational grid points. As a result, the speed-up of the reduced order model cannot be expected to be as dramatic.

The method proposed in this work aims at alleviating this disadvantage of the subspace projection by constructing the reduced order model such that it does not require the evaluation of the residual at each and every grid point. Rather, a projection is proposed such that the reduced order model makes use of the residual at only few of the grid points. In this way, independence from the full order dimension is achieved, while keeping the flexibility of the subspace projection approach. Hence, the proposed method incorporates the benefits of both the Galerkin and the subspace projection.

The approach of evaluating the spatial discretization on a subset of the grid points is inspired by recent advances in nonlinear model order reduction techniques. Both missing point estimation (MPE) [5] and the discrete empirical interpolation method (DEIM) [13] make use of this idea, but have not been applied to aerodynamic applications before.

The difference between both methods is that DEIM computes a second POD basis for the nonlinear right hand side. For this, the right hand sides of the snapshots are used. Note that due to the fact that these will be zero for steady flow problems, the method will fail for such problems.

1.1.2 The least-squares ROM

Another model order reduction approach designed for aerodynamics is the so-called least-squares reduced order model. It is also based on proper orthogonal decomposition, but not on projection. The basic idea is to find a solution in the POD subspace, which has the minimal residual in a least-squares sense for a given flow condition. Due to the fact that solutions are sought in the POD subspace, the resulting reduced order model is low-dimensional. In fact, only suitable coefficients for the POD basis vectors have to be found.

The method has been introduced by LeGresley and Alonso [36, 37] and has since been used by other authors [73, 72, 11]. It has been successfully applied to an airfoil design problem [36] and to predict transonic flow fields of two-dimensional airfoils [37] as well as three-dimensional configurations [73].

Unlike the previously introduced methods, the least-squares ROM is not based on projecting the governing equations onto the POD subspace. Instead, the approach fits into the framework of weighted residuals. As the formulated problem of finding the solution with the minimal residual is an optimization problem, it is solved with a Newton-type of method.

1.2 Goal of this work

The goal of this work is to implement a projection-based reduced order model, which is capable of efficiently simulating steady flows around industrially relevant two-dimensional airfoils and complex three-dimensional airplane configurations. The steady states for these test cases shall be computed for different flow conditions, which are controlled through parameters such as the angle of attack and the onflow velocity (see Section 2.4) in the governing equations.

In the construction of the reduced order model (ROM) the following properties of the ROM shall be regarded:

1. Independence from the full order dimension of the original CFD model shall be achieved by setting up a suitable projection such that the residuals have to be evaluated only at a subset of all possible grid locations.

2. The reduced order model shall use state-of-the-art simulation techniques including turbulence models and shall be applicable to realistic industrial test cases.

3. User interaction should be kept to a minimum such that the ROM is as user-friendly as possible.

Validation of the implemented model will be done by comparing relevant outputs such as the pressure and force coefficients - which will be introduced later on in this thesis – to the original CFD model output. As a reference CFD solver, the numerical software TAU [28, 22, 21] of the German Aerospace Center (DLR) is used. TAU is also utilized in the reduced model order to evaluate the residual.

1.3 Overview of this work

This work is subdivided into three parts. These parts structure the thesis thematically. The first part is concerned with computational fluid dynamics. In it, fundamentals of fluid mechanics are established, the governing equations of fluid mechanics are derived, and numerical methods for solving these equations are presented. In the next part model order reduction is described. After some fundamentals and an overview are given, the methods used in this thesis, the proper orthogonal decomposition and missing point estimation, are outlined. Finally, in the last part the model order reduction techniques of Part II are applied to the discrete governing equations of fluid dynamics derived in Part I. In the following the chapters are described individually.

Part I: After this introductory chapter to the thesis, the purpose of the next one is to give the reader some background on fluid dynamics. In particular, in **Chapter 2** important definitions of concepts like fluids, the flow conditions and the aerodynamic coefficients are stated.

In **Chapter 3** the governing equations of fluid dynamics are derived. Furthermore, the difference between the Euler and the Navier-Stokes equations as well as steady and unsteady flows are discussed.

The governing equations are then discretized in **Chapter 4**. All major steps in the discretization, namely grid generation, spatial and temporal discretization, are outlined. At the end of the chapter, the initial and boundary conditions, which are used in CFD are presented and some remarks on the non-dimensionalization of the physical quantities are given.

Part II: The following four chapters belong to Part II, which is concerned with model order reduction (MOR). The first of which, **Chapter 5**, provides the reader with useful mathematical background. In fact, the singular value decomposition and projections are defined and important properties are given.

Chapter 6 then gives an overview of how projection-based model order reduction works. It describes the two basics principles of MOR, namely approximation in a low-dimensional subspace and approximation by projection.

The first principle is achieved in this work by proper orthogonal decomposition (POD). POD is outlined in **Chapter 7** both for the discrete Euclidean space as well as for general Hilbert spaces. In the end of the chapter, some remarks on affine POD implementations are given.

The second basic principle of model order reduction, which is the approximation by projection, is implemented in this work by a method called missing point estimation (MPE). It is described in **Chapter 8**. In the different sections some comments on the projection and the point selection as it can be found in literature are given. Additionally, this chapter contains a new error estimation for the method, which is based on the error estimation of a similar technique called the discrete empirical interpolation method (DEIM). With this chapter Part II is completed and the last part begins.

Part III: First of all, in **Chapter 9** the numerical implementation of the missing point estimation approach to aerodynamic applications is described in detail. The description presents the numerical methods as well as in the last section how the methods are realized in the code.

Subsequently, numerical results for the missing point estimation are given in **Chapter 10**. While in the tests of this chapter only the angle of attack is a varying system parameter, in the next one also the Mach number is a parameter. In this way some limitations of the proposed method can be devised in **Chapter 11**. Furthermore, relevant norms of the right hand side of the MPE system are analyzed and thus important properties of the method are presented.

At last, **Chapter 12** concludes the thesis with a summary of the findings and an outlook on future work, which can be done.

Part I

Computational fluid dynamics

Chapter 2

Fundamental concepts of fluid mechanics

In this chapter some fundamental concepts of fluid dynamics are presented. The chapter is based on the textbooks [7, 16, 20, 34, 69].

When dealing with fluid mechanics for the first time, a natural question to ask is how a fluid is defined. This question will be answered in the first section. Next, the continuum hypothesis is given. It states that since in fluid mechanics one is interested in the overall properties of the fluid, it suffices to consider fluids as varying continuously. This hypothesis is used in the modeling and the numerical methods, presented in Part I. In Section 2.3 stresses are addressed, which are needed for modeling the Navier-Stokes equations. The Navier-Stokes equations describe the behavior of viscous flows. Before explaining in Section 2.5 what viscous flows are – along with three other characterizations of flows, namely turbulence, compressibility and the flow regime – two important parameters, the Mach number and the angle of attack, are defined. These parameters describe the flow condition. Finally, in Sections 2.6 and 2.7 the definitions of the pressure and force coefficient are given, which are non-dimensional quantities that characterize aircrafts.

2.1 What is a fluid?

In fluid dynamics a distinction of the state of substances is made between *solids* and *fluids*. The difference between the two states lies in their resistance to a *shear force*, which is a force acting tangential to the surface of the substance. Solid materials are able to resist moderate shear stresses by deforming only slightly. When the stress is relieved, perfectly elastic solids regain their original state. This is not true for fluids. They deform continuously under the exertion of a shear force to it, no matter how small this force may be, and do not regain a preferred shape. In colloquial language this is referred to as their ability to *flow* [7, 16, 20, 34, 69].

Note that the distinction between fluids and solids is not a sharp one, but is gradual. Honey for example does not deform as easily as water, but it is definitely not a solid either.

The different behavior of the state of substances to shear stresses is caused by the intensity of the *intermolecular forces* holding the molecules of the materials together. Fluids have less intermolecular forces than solids, resulting in their ability to flow. Related to these forces are two different classes of fluids, namely *gases* and *liquids*. The former have less molecular forces than the later. A prominent example of a gas is air, whereas one of liquids is water [7].

2.2 Continuum hypothesis

When investigating fluids on a microscopic scale, they are discontinuous. This is due to the fact that the mass of fluids is not continuously distributed in space, but is concentrated in molecules [20, p.20]. However, in fluid mechanics in general the interest lies in the overall behavior of a fluid on a macroscopic scale rather than in the behavior of the discrete molecules [7, p.4]. For example, when studying the forces exerted on the boundaries of an object by a gas like air, it is insignificant which exact molecule collides with the boundary creating a small force, but rather the *pressure*, which is the statistical average of the forces per unit area, is of interest [34, p.5].

As a result, fluids are considered to be a *continuum* on a macroscopic scale, i.e. they are assumed to vary smoothly in space. This is the so-called *Continuum Hypothesis* which is a fundamental concept of classical fluid dynamics. This hypothesis holds true as long as there are a vast amount of molecules, even in a small volume. For a more precise definition of the validity of the continuum hypothesis the Knudsen number is used. The reader is referred to [25, p.13] for specifics on its definition.

2.3 Stresses

When a fluid moves along a body, it will experience *surface forces* like friction. The concept of *stress* describes how these forces are transported through the fluid [20, p.27]. In the following a formal definition of stress will be given.

Consider a point in an orthogonal coordinate system through which a fluid moves and the plane passing through the point with its normal vector in direction of the x-axis. Let δA_x be a small region of this plane in the neighborhood of the point. Further let $\delta \boldsymbol{f} = \begin{bmatrix} \delta f_x & \delta f_y & \delta f_z \end{bmatrix}^T$ be a force acting on δA_x. Then the stress components for δA_x are defined as

$$\begin{aligned}
\tau_{xx} &= \lim_{\delta A_x \to 0} \frac{\delta f_x}{\delta A_x}, \\
\tau_{xy} &= \lim_{\delta A_x \to 0} \frac{\delta f_y}{\delta A_x}, \\
\tau_{xz} &= \lim_{\delta A_x \to 0} \frac{\delta f_z}{\delta A_x}.
\end{aligned}$$

Note that τ_{xx} is a *normal stress*, whereas τ_{xy} and τ_{xz} are *shear stresses*. The three stresses are labeled with two subscripts: the first to indicate the plane and the second to indicate the direction in which the stress acts [20, p.29].

The stress components τ_{yy}, τ_{yx}, τ_{yz}, τ_{zz}, τ_{zx}, τ_{zy}, corresponding to the planes whose normal are in direction of the y- and the z-axis, are defined accordingly. The stress at a point is described by all nine stress components which are stored in a compact form in the *stress tensor*

$$\overline{\overline{\tau}} = \begin{pmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \end{pmatrix}.$$

2.4 Flow conditions: the Mach number and angle of attack

In the process of developing an airfoil or airplane, the aerodynamic behavior at different flow conditions are of interest. Such flow conditions are for example the velocity of the aerodynamic body and the onflow angle. The latter is the angle at which the air hits the aerodynamic body. Obviously, both the velocity and the onflow angle will be very different during take-off, cruise and landing.

Two parameters associated with the flow conditions, namely the *Mach number* and the *angle of attack*, shall be defined next. Both parameters influence the boundary and possibly also the initial conditions of the governing equations, as it will be shown in Chapter 4 in equation (4.9).

The *Mach number* indicates how fast the flow is in relation to the speed of sound. It is defined as

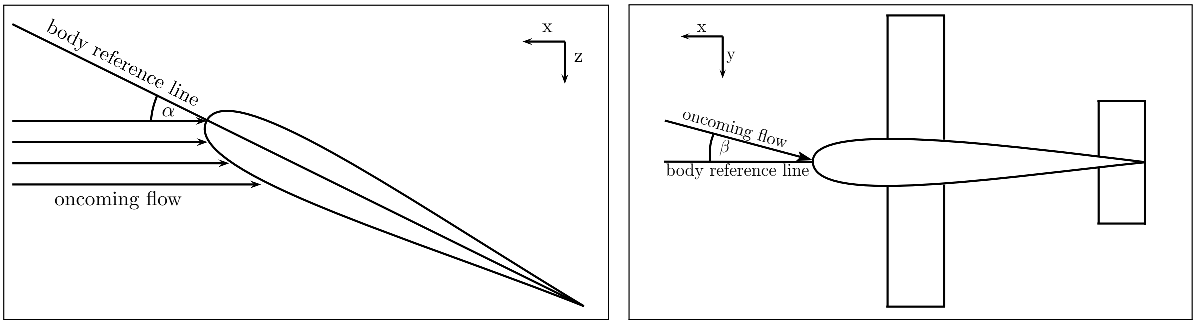
$$M = \frac{|\boldsymbol{v}|}{a},$$

where $\boldsymbol{v}$ is the velocity vector of the undisturbed onflow and a is the speed of sound. The undisturbed onflow is often called freestream and $\boldsymbol{v}$ is thus also referred to as the freestream velocity vector. The speed of sound for air – under the assumption that air is a perfect gas – is calculated by $a = \sqrt{\gamma R T_\infty}$ with the *adiabatic constant* γ, the *gas constant* R as well as the freestream temperature T_∞. The speed of sound for air is approximately $343\frac{m}{s}$ at $20°C(= 293.15\,K)$.

The *angle of attack* is defined as the angle between the oncoming flow and the body reference line. For two dimensions this is portrayed for an airfoil in Figure 2.1(a). While in two dimensions the onflow angle usually has only a vertical component α, in three dimensions a second angle β, the *sideslip angle* or *yaw angle*, is used to describe the horizontal component of the onflow angle. The sideslip angle is illustrated in Figure 2.1(b). Note that the x-y-plane stands for the horizontal plane.

2.5 Characterization of flows

Depending on what kind of flow is considered, flows can be very different in nature. The following characterizations describe flows.



(a) The angle of attack. (b) The sideslip angle.

Figure 2.1: *The onflow angles.*

2.5.1 Viscous and inviscid flows

Objects which are moving through air such as an airplane experience aerodynamic drag. Drag is a force acting in opposite direction to the motion of the airplane (see Figure 2.3). A valid question to ask is: Where does this drag come from? One component of the drag is due to friction of the air as it moves along the object [20, p.38].[1] This essentially is *viscosity*. Note that a flow is called *inviscid*, if it is assumed to have no viscosity, which is an idealization. Otherwise, it is named *viscous*.

The influence of viscosity is characterized by the *Reynolds number*. It measures the ratio between inertial and viscous forces and is defined by

$$Re = \frac{|\boldsymbol{v}| l \rho}{\mu}, \tag{2.1}$$

where l is the characteristic size scale of the flow (e.g. the length of an airfoil), v its velocity, ρ its density and μ the dynamic viscosity [20, p.38]. Note that inertia is the tendency of an object to resist a change in its motion.

The smaller the Reynolds number, the more significant is the role played by the viscous force on the aerodynamic drag. But even for high-Reynolds number flows Prandtl stated that there is a small region near the surface of the body, called the *boundary layer*, where friction is significant. In the boundary layer the velocity rises quickly from zero at the surface of the object to the value predicted by inviscid flow theory which marks the end of the boundary layer [20, p.39]. This is illustrated in Figure 2.2.

2.5.2 Laminar and turbulent flows

Laminar flows are characterized by their fluid particles moving in smooth layers. On the other hand in *turbulent flows* the fluid particles mix as they move [20, p.40]. Due to small scale fluctuations turbulent flows are difficult to analyze and to simulate numerically.

A dimensionless number indicating whether flows are turbulent or not is the *Reynolds number*. It measures the ratio between inertia and viscous forces and is defined by (2.1). Below a Reynolds

---

[1] Other components of drag are the pressure drag, induced drag, wave drag etc. [27, p.35].

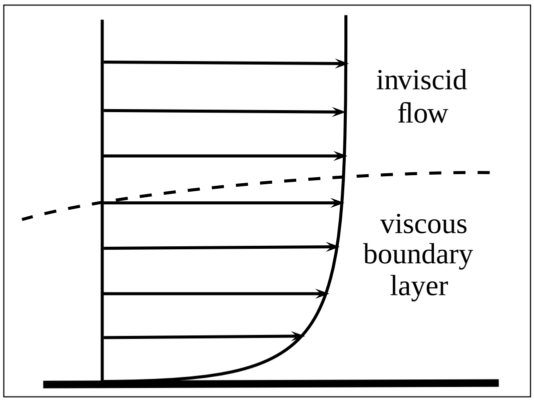


Figure 2.2: *Sketch of a boundary layer [20, p.39].*

number of about $Re = 2300$ a flow is laminar [20, p.304]. The Reynolds number for aerodynamic applications is typically very high. For aircrafts, it is typically in the order of 10^7.

2.5.3 Compressible and incompressible flows

If the density of a fluid varies significantly (due to a change in pressure), then the fluid is compressed in those regions, where the density is high. As a result, the corresponding flow is called *compressible*. The most important property of compressible flows is the possible occurrence of shock waves.

If, on the other hand, the variation of density is negligible, a flow is called *incompressible* [20, p.41]. Note that incompressible flows do not actually exist since there are always some changes in density. However, as their definition implies these changes are irrelevant for the analysis of them.

Most liquid flows at modest pressures are incompressible, while gas flows are compressible [20, p. 41]. However, there are exceptions. For example gas flows at low speeds and with insignificantly small heat transfer are incompressible [20, p. 42]. In airplane aerodynamics flows are usually compressible.

2.5.4 The flow regimes: subsonic, transonic, and supersonic flow

As a final characterization of flows, the flow regime is described. The flow regimes of airplanes can be categorized into *subsonic*, *transonic*, and *supersonic* flows. Other regimes such as hypersonic and re-entry flows will not be of interest for this thesis.

The literal meaning of subsonic and supersonic is that the velocity is below and above the speed of sound, respectively. In aerodynamics, however, the flow regimes are used to describe a particular range of Mach numbers. In this respect the *critical Mach number* is important. It is the freestream Mach number M_∞ for which a local Mach number of one is reached for the first time somewhere on the surface. Flows for which the freestream Mach number is below the critical Mach number, are called *subsonic*. Above the critical Mach number the flow is *transonic*, i.e. the flow is subsonic in some regions and *supersonic* in other regions, until at a Mach number of around 1.2, where most parts of the flow are supersonic. In this latter case the flow is called *supersonic* [27].

2.6 The pressure coefficient

Pressure distribution on the surface is important in the analysis of the behavior of aircrafts. Since it is convenient to deal with dimensionless quantities, the *pressure coefficient* c_p is used. It is given by

$$c_p = \frac{p - p_\infty}{q_\infty}, \tag{2.2}$$

where $q_\infty = \frac{1}{2}\rho_\infty |\boldsymbol{v}_\infty|^2$ is called the dynamic pressure. Note that p stands for the pressure at a point, p_∞ for the freestream pressure, $\boldsymbol{v}_\infty$ for the freestream velocity and ρ_∞ for the freestream density.

In [1, p. 539] the alternative definition of the form

$$c_p = \frac{2}{\gamma M_\infty^2}\left(\frac{p}{p_\infty} - 1\right)$$

is derived. It is equivalent to (2.2) and is expressed in terms of the freestream Mach number M_∞ instead of the dynamic pressure. Note that the γ stands for the ratio of specific heat coefficients at constant pressure and volume.

2.7 Force coefficients

When air is flowing past a body such as an airplane, the air will change its original path. In fact, the speed of the air will be different in the different sections of the airplane. This as well as the viscosity will lead to forces that are exerted on the plane [27, p.26]. The force is separated into three perpendicular components – one for each spatial dimension. In Figure 2.3 these forces as well as the moments are illustrated. The direction of the three components is determined by the direction of flight, which corresponds to the x-axis.

Before some of the components are described in detail, the force coefficient will be defined in general. The force coefficient is a dimensionless quantity. Assume that $\boldsymbol{F}$ is an aerodynamic force, then the dynamic pressure $\frac{1}{2}\rho_\infty |\boldsymbol{v}_\infty|^2$ multiplied by the surface area S, which the force acts upon, is used for non-dimensionalization. Thus, the force coefficient is given by

$$c_{\boldsymbol{F}} = \frac{|\boldsymbol{F}|}{\frac{1}{2}\rho_\infty |\boldsymbol{v}_\infty|^2 S}. \tag{2.3}$$

The two most important aerodynamic force coefficients, the lift and drag coefficient, will be explained next.

2.7.1 The lift force and lift coefficient

The lift is the component of force, perpendicular to the direction of flight, which acts upwards. What is meant with upwards is that the “pilot’s head is above the feet” [27, p.26]. However,

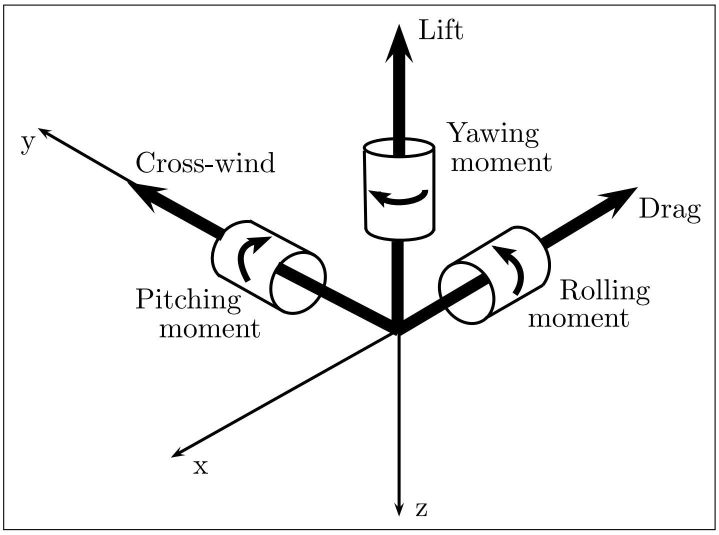
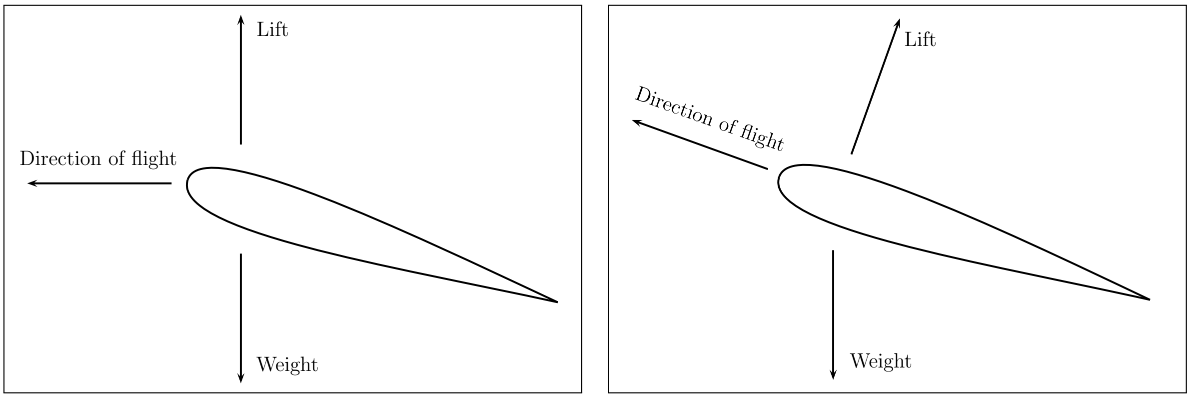


Figure 2.3: *The forces and moments [27, p.28].*

note that "upwards is not fixed relative to the aircraft" [27, p.26]. In fact, the upwards direction is different for a plane flying at a high angle of attack (see Figure 2.4(a)), for example in the take-off phase, and a plane in climbing flight (Figure 2.4(b)).



(a) Flight at a high angle of attack.

(b) Climbing flight.

Figure 2.4: *The direction of the lift force.*

According to (2.3) the *lift coefficient* is defined as

$$c_l = \frac{l}{\frac{1}{2}\rho_\infty |\boldsymbol{v}_\infty|^2 S}. \tag{2.4}$$

Note that l denotes the magnitude of the lift force.

2.7.2 The drag force and drag coefficient

The drag is the component of force acting in the same direction as the oncoming flow. In other words, it is exerted in the opposite direction of the motion of the airplane. This justifies the name of this component of force.

With the definition of drag and with (2.3) the drag coefficient is given by

$$c_d = \frac{d}{\frac{1}{2}\rho_\infty |\boldsymbol{v}_\infty|^2 S}. \tag{2.5}$$

Note that d denotes the magnitude of the drag force.

Chapter 3

Derivation of the governing equations

The derivation of the governing equations of fluid dynamics is based on conservation of mass, momentum and energy. These conservation laws are very intuitive and are well known from elementary physics. For example, the law of mass conservation states that mass can neither be created nor destroyed, which the reader might know from school physics.

Before the conservation laws are presented in detail in the following sections, an overview of how they are formulated shall be given here. In particular, the way of mathematically describing the flow and its physical quantities shall be discussed.

The most intuitive way of describing the motion of a fluid is by describing the trajectory of each particle. This is referred to as the *Lagrangian description* of a flow [17, p.15]. In a similar fashion a number of particles can be considered at once, which constitute the so-called *control mass* [18, p.3]. While it is straight forward to identify the control mass for solid bodies, this is not the case for fluids like air, since the particles will mix with time and the control mass might eventually be separated into disjoint domains.

This gives rise to another way of describing a flow, which – instead of looking at a fixed set of particles – considers a spatial domain, called *control volume*, and determines the velocity of the particles passing through the domain. This is called the *Eulerian description* [17, p.15] of a flow or *control volume approach.*

In order to be able to formulate the conservation laws in control volume form, the control mass approach is presented first, succeeded by a means of transforming this approach into the control volume formulation:

Consider the physical quantities mass, momentum and energy for a control mass. Note that these are *extensive properties* [18, p. 3], that is, they depend on the amount of matter, which is regarded. In the following such an extensive property Ψ is measured by integrating the product of an *intensive property* ϕ and the density ρ over the domain Ω_{CM} occupied by the control mass, that is,

$$\Psi = \int\limits_{\Omega_{CM}} \rho\,\phi\,\mathrm{d}\Omega_{CM}. \tag{3.1}$$

Intensive properties ϕ are independent of the amount of matter considered and are either vectors or scalars [18, p. 3]. In the following sections it can be observed that ϕ is equal to 1, to the vector of Cartesian velocities $\boldsymbol{v}$ and to the mass-specific total Energy E for the extensive properties mass, momentum and energy, respectively.

All conservation laws have in common that they make statements on the rate of change of the property at hand. For example the mass conservation law states that mass is neither created nor destroyed. This means that the rate of change of mass is zero. Translated into a mathematical statement, this reads

$$\frac{\mathrm{dm}}{\mathrm{d}t} = 0,$$

where m stands for mass and t for time [18, p. 3].

Note that the rate of change of an extensive property is in fact its derivative with respect to time. Since the extensive properties in (3.1) are given by integrals, the derivatives of these integrals bound on the control mass have to be considered. The control volume equation, which is stated in *Reynolds transport theorem*,[1] is a means to transform these derivatives to a control volume formulation and is as follows

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega_{CM}} \rho\,\phi\,\mathrm{d}\Omega_{CM} = \int\limits_{\Omega} \frac{\partial(\rho\,\phi)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_{S} \rho\,\phi\,(\boldsymbol{v}^T \cdot \boldsymbol{n})\,\mathrm{d}S, \tag{3.2}$$

where Ω, S and Ω_{CM} stand for the control volume, the surface of the control volume and the volume in which the control mass (CM) lies, respectively. The vector $\boldsymbol{n}$ denotes the outer facing normal vector of the surface of the control volume S and $\boldsymbol{v}$ is the vector of Cartesian velocities. Equation (3.2) states that the rate of change of an extensive property of the control mass is equal to the sum of the contributions due to the rate of change inside the control volume and due to the flux – that is the amount of the property entering or leaving the control volume in normal direction to the surface [18, p.4]. The last term is referred to as the *convective flux*.

The following chapter is based on the lectures of C.-C. Rossow [51] and on the textbooks [10, 35, 18]. Note that all vectors as well as all vector-valued functions will be written in bold letters. All scalars, vectors, and functions are real-valued.

3.1 Mass conservation

At first the conservation law for the extensive property mass will be considered. Note that *density* ρ is defined as mass m per unit volume V_0: $\rho = \mathrm{m}/V_0$. Therefore the mass inside a volume occupied by the control mass is

$$\mathrm{m} = \int\limits_{\Omega_{CM}} \rho\,\mathrm{d}\Omega_{CM}. \tag{3.3}$$

---

[1] Refer to [20, Chapter 4] for a physical motivation and to [17, p. 19] for a rigorous proof of the theorem. Note that in order to proof the transport theorem continuous differentiability of the intensive property at hand along with other smoothness conditions are needed.

By comparing equations (3.1) and (3.3) it follows that the intensive property is $\phi = 1$. The mass conservation law states that mass is neither created nor destroyed. In other words, the rate of change of mass is zero at all times:

$$\frac{\mathrm{dm}}{\mathrm{d}t} = 0. \tag{3.4}$$

Inserting (3.3) into equation (3.4) and using (3.2) yields

$$\int\limits_{\Omega} \frac{\partial \rho}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} \rho \, (\boldsymbol{v}^T\!\cdot \boldsymbol{n})\, \mathrm{d}S = 0. \tag{3.5}$$

This equation is called the mass conservation equation.

3.2 Momentum balance

Momentum is defined by the product of mass m and the velocity $\boldsymbol{v}$: $\boldsymbol{I} = \mathrm{m} \cdot \boldsymbol{v}$. In a volume of control mass this translates to

$$\boldsymbol{I} = \int\limits_{\Omega_{CM}} \rho\, \boldsymbol{v}\, \mathrm{d}\Omega_{CM}. \tag{3.6}$$

Note that the intensive property is $\phi = \boldsymbol{v}$. Using equation (3.2), the temporal rate of change of momentum becomes

$$\frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}t} = \int\limits_{\Omega} \frac{\partial (\rho\, \boldsymbol{v})}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} (\rho\, \boldsymbol{v})\, (\boldsymbol{v}^T\!\cdot \boldsymbol{n})\, \mathrm{d}S. \tag{3.7}$$

According to Newton's second law of motion the temporal rate of change of momentum $\frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}$ corresponds to the sum of all forces $\sum \boldsymbol{F}$ acting on the control mass

$$\frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}t} = \sum \boldsymbol{F}. \tag{3.8}$$

The forces acting on the control mass can be:

- body forces
- surface forces.

Body forces are for example gravity, magnetic and electromagnetic forces. They are summed up to

$$\boldsymbol{F}_b = \int\limits_{\Omega} \rho\, \boldsymbol{f_b}\, \mathrm{d}\Omega.$$

The surface forces are e.g. pressure as well as shear stresses and are expressed by

$$\boldsymbol{F}_S = -\oint\limits_{S} p\, \boldsymbol{n}\, \mathrm{d}S + \oint\limits_{S} \overline{\overline{\tau}}\, \boldsymbol{n}\, \mathrm{d}S.$$

Note that there is a minus sign in front of the first term, since pressure in opposite direction to the normal vector $\boldsymbol{n}$ shall be positively measured. The second term in the above equation represents viscous forces tangential to the surface. The viscous stress tensor is denoted by

$$\overline{\overline{\tau}} = \begin{pmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \end{pmatrix}.$$

With the definition of the forces and equation (3.7) the momentum balance equation (3.8) becomes

$$\int\limits_{\Omega} \frac{\partial(\rho\,\boldsymbol{v})}{\partial t}\,\mathrm{d}\Omega + \oint\limits_{S} \rho\,\boldsymbol{v}\,(\boldsymbol{v}^{T}\!\cdot\boldsymbol{n})\,\mathrm{d}S + \oint\limits_{S} p\,\boldsymbol{n}\,\mathrm{d}S - \oint\limits_{S} \overline{\overline{\tau}}\,\boldsymbol{n}\,\mathrm{d}S = \int\limits_{\Omega} \rho\,\boldsymbol{f_b}\,\mathrm{d}\Omega. \tag{3.9}$$

Note that the above equation (3.9) is not a scalar, but a vector equation. The dimension of these vectors is typically either two or three, depending on the spatial dimension of the considered problem. For example, if a two-dimensional airfoil is considered, then (3.9) consists of two equations. On the other hand, if a three-dimensional aircraft configuration is investigated, then (3.9) stands for three equations. Take a note of the fact that the body forces $\boldsymbol{F}_b$ are often neglected in aerodynamics, i.e. the right hand side of (3.9) is zero.

3.3 Energy balance

The total energy in a volume is defined by

$$E_T = \int\limits_{\Omega_{CM}} \rho\,E\,\mathrm{d}\Omega_{CM}, \tag{3.10}$$

where E is the mass-specific energy

$$E = e + \frac{|\boldsymbol{v}|^2}{2}.$$

In this formula e represents the mass-specific inner energy and $\frac{|\boldsymbol{v}|^2}{2}$ the mass-specific kinetic energy. The potential energy of gases will be neglected.

Note that $\phi = E$ is the intensive property. Using equation (3.2) the temporal rate of change of energy becomes

$$\frac{\mathrm{d}E_T}{\mathrm{d}t} = \int\limits_{\Omega} \frac{\partial(\rho\,E)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_{S} (\rho\,E)\,(\boldsymbol{v}^{T}\!\cdot\boldsymbol{n})\,\mathrm{d}S. \tag{3.11}$$

According to the law of energy conservation, energy is neither created nor destroyed. It is only transformed from one form to another. More precisely, the first law of thermodynamics states that the change in the total energy is equal to the sum of the heat transfer $\dot{Q}$ into the control volume and the power $\dot{W}$ delivered in the volume:

$$\frac{\mathrm{d}E_T}{\mathrm{d}t} = \dot{Q} + \dot{W}. \tag{3.12}$$

The heat transfer $\dot{Q}$ consists of the diffusive heat flux, representing the diffusion of heat due to molecular conduction, and the volumetric heating caused by emission or absorption of heat radiation. The former is given by

$$\dot{Q}_d = -\oint\limits_S \boldsymbol{K}^T \cdot \boldsymbol{n}\, \mathrm{d}S,$$

where the heat transfer vector is

$$\boldsymbol{K}^T = \begin{bmatrix} -k\frac{\partial T}{\partial x} & -k\frac{\partial T}{\partial y} & -k\frac{\partial T}{\partial z} \end{bmatrix}$$

with the temperature T and the heat transfer coefficient k. The volumetric heating is represented by

$$\dot{Q}_V = \int\limits_\Omega \dot{q}_h \mathrm{d}\Omega,$$

where $\dot{q}_h$ stands for the rate of heat transfer per unit mass.

In the following the different powers, which are delivered in the volume, will be listed:

- power of the pressure acting in opposite direction of the normal vector

$$\dot{W}_p = -\oint\limits_S p\,(\boldsymbol{v}^T \cdot \boldsymbol{n})\, \mathrm{d}S$$

- power of the viscous forces

$$\dot{W}_v = \oint\limits_S \boldsymbol{v}^T \cdot \left(\overline{\overline{\tau}} \cdot \boldsymbol{n}\right)\, \mathrm{d}S$$

- power of the body forces

$$\dot{W}_{f_b} = \int\limits_\Omega \rho\, \boldsymbol{f_b}^T \cdot \boldsymbol{v} \mathrm{d}\Omega$$

Inserting the definition of $\dot{Q} = \dot{Q}_d + \dot{Q}_V$ and $\dot{W} = \dot{W}_p + \dot{W}_v + \dot{W}_{f_b}$ into equation (3.12) yields

$$\begin{aligned} \int\limits_\Omega \frac{\partial(\rho E)}{\partial t}\, \mathrm{d}\Omega \quad &+ \quad \oint\limits_S (\rho E)\,(\boldsymbol{v}^T \cdot \boldsymbol{n})\, \mathrm{d}S + \oint\limits_S p\,(\boldsymbol{v}^T \cdot \boldsymbol{n})\, \mathrm{d}S - \int\limits_\Omega \left[\rho\,(\boldsymbol{f_b}^T \cdot \boldsymbol{v}) + \dot{q}_h\right]\, \mathrm{d}\Omega \\ &= \quad \oint\limits_S \boldsymbol{v}^T \cdot \left(\overline{\overline{\tau}} \cdot \boldsymbol{n}\right)\, \mathrm{d}S - \oint\limits_S (\boldsymbol{K}^T \cdot \boldsymbol{n})\, \mathrm{d}S. \end{aligned} \tag{3.13}$$

By defining the mass-specific total enthalpy

$$H = E + \frac{p}{\rho}$$

this equation can be further simplified to

$$\begin{aligned} \int\limits_\Omega \frac{\partial(\rho E)}{\partial t}\, \mathrm{d}\Omega \quad &+ \quad \oint\limits_S (\rho H)\,(\boldsymbol{v}^T \cdot \boldsymbol{n})\, \mathrm{d}S - \oint\limits_S \boldsymbol{v}^T \cdot \left(\overline{\overline{\tau}} \cdot \boldsymbol{n}\right)\, \mathrm{d}S + \oint\limits_S (\boldsymbol{K}^T \cdot \boldsymbol{n})\, \mathrm{d}S \\ &= \quad \int\limits_\Omega \left[\rho\,(\boldsymbol{f_b}^T \cdot \boldsymbol{v}) + \dot{q}_h\right]\, \mathrm{d}\Omega. \end{aligned} \tag{3.14}$$

Again, note that in aerodynamics the body forces as well as heat radiation are often neglected, i.e. $\boldsymbol{f_b} = \boldsymbol{0}$ and $\dot{q}_h = 0$.

3.4 Additionally needed equations for a perfect gas formulation

Note that if body forces as well as heat radiation are neglected, the system of the equations (3.5), (3.9) and (3.13) contains seven unknown flow field variables (ρ, v_x, v_y, v_z, p, E, T), but there are only five equations. In order to be able to solve the system, in the following subsection two thermodynamic relations (3.15) and (3.17) are used, which relate the variables to one another. In addition, the viscosity coefficient μ and the heat transfer coefficient k are needed to obtain the viscous stresses $\overline{\overline{\tau}}$ and the heat transfer vector $\boldsymbol{K}$.

3.4.1 Thermodynamic relations

In aerodynamics it is assumed that the air behaves like a *perfect gases*, for which

$$p = \rho R T \tag{3.15}$$

holds, where R is the *gas constant*.[2] It is defined by $R = c_p - c_v$ with the *specific heat at constant pressure* c_p and the *specific heat at constant volume* c_v. For perfect gases it is assumed that c_p as well as c_v are constant. Consequently, the *ratio of specific heats* $\gamma = c_p/c_v$ is also constant and is $\gamma = 1.4$ for air [17, p. 29]. The internal energy of a perfect gas is defined by

$$e = c_v T. \tag{3.16}$$

Insertion of $T = e/c_v$ into (3.15) yields the so-called *equation of state*

$$p = \rho(\gamma - 1)e. \tag{3.17}$$

3.4.2 Viscous stresses

The following description of viscous stresses is taken from [35, p.24]. For *Newtonian fluids* it is assumed that the entries of the stress tensor $\overline{\overline{\tau}}$ are given by

$$\begin{aligned}
\tau_{xx} &= 2\mu\frac{\partial v_x}{\partial x} - \frac{2}{3}\mu\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right),\\
\tau_{yy} &= 2\mu\frac{\partial v_y}{\partial y} - \frac{2}{3}\mu\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right),\\
\tau_{zz} &= 2\mu\frac{\partial v_z}{\partial z} - \frac{2}{3}\mu\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right),\\
\tau_{xy} &= \mu\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right) = \tau_{yx},\\
\tau_{xz} &= \mu\left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right) = \tau_{zx},\\
\tau_{yz} &= \mu\left(\frac{\partial v_y}{\partial z} + \frac{\partial v_z}{\partial y}\right) = \tau_{zy},
\end{aligned}$$

---

[2] For air, $R = 287 \frac{N \cdot m}{kg \cdot K}$.

where μ is the *coefficient of shear viscosity.* It is strongly dependent on temperature, but only weakly on pressure, and it is approximated by the *Sutherland formula* [69, pp.27]

$$\mu = \mu_0 \frac{T_0 + c}{T + c} \left(\frac{T}{T_0} \right)^{3/2}, \tag{3.18}$$

where μ_0 is the viscosity measured at a reference temperature T_0, and c is the Sutherland's constant. The values of these parameters are found empirically and are approximately $\mu_0 = 1.71 \cdot 10^{-5}$, $T_0 = 273$, and $c = 110.4$ [69, p.771]. After inserting these values, the Sutherland formula reads

$$\mu = 1.453 \cdot 10^{-6} \cdot \left(\frac{T^{3/2}}{T + 110.4} \right).$$

3.4.3 Heat transfer coefficient

The heat transfer coefficient k for air is given by

$$k = c_p \frac{\mu}{P_r},$$

where P_r is the Prandtl number which is assumed to be constant in the entire flow field with a value of $P_r = 0.72$ for air [10, p. 19].

3.5 The governing equations in integral form

Combining the conservation equations (3.5), (3.9) and (3.14) into one set of equations yields

$$\begin{aligned}
&\int\limits_{\Omega} \tfrac{\partial \rho}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} \rho\, (\boldsymbol{v}^{T}\!\cdot \boldsymbol{n})\, \mathrm{d}S = 0 \\
&\int\limits_{\Omega} \tfrac{\partial (\rho\, \boldsymbol{v})}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} \rho\, \boldsymbol{v}\, (\boldsymbol{v}^{T}\!\cdot \boldsymbol{n})\, \mathrm{d}S + \oint\limits_{S} p\, \boldsymbol{n}\, \mathrm{d}S - \oint\limits_{S} \overline{\overline{\tau}}\, \boldsymbol{n}\, \mathrm{d}S = \int\limits_{\Omega} \rho\, \boldsymbol{f_b}\, \mathrm{d}\Omega \\
&\int\limits_{\Omega} \tfrac{\partial (\rho\, E)}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} \rho\, H\, (\boldsymbol{v}^{T}\!\cdot \boldsymbol{n})\, \mathrm{d}S - \oint\limits_{S} \boldsymbol{v}^{T}\!\cdot \left(\overline{\overline{\tau}} \cdot \boldsymbol{n} \right)\, \mathrm{d}S + \oint\limits_{S} \boldsymbol{K}^{T}\!\cdot \boldsymbol{n}\, \mathrm{d}S = \int\limits_{\Omega} \left[\rho\, (\boldsymbol{f_b}^{T}\!\cdot \boldsymbol{v}) + \dot{q}_h \right] \mathrm{d}\Omega.
\end{aligned}$$

In the following the goal is to represent the equations in a shorter form. For this purpose the *vector of the conservative variables* $\boldsymbol{W}$, the *flux tensor* $\overline{\overline{F}}$ and the *source term* $\boldsymbol{Q}$ will be defined:

Firstly, all variables in the above set of equations, which are differentiated with respect to time t, will be collected in a vector. This vector will be called the *vector of the conservative variables* and reads

$$\boldsymbol{W} = \begin{bmatrix} \rho \\ \rho\, v_x \\ \rho\, v_y \\ \rho\, v_z \\ \rho\, E \end{bmatrix}. \tag{3.19}$$

Secondly, the *flux tensor* $\overline{\overline{F}}$ is defined by

$$\overline{\overline{F}} = \overline{\overline{F}}_c - \overline{\overline{F}}_v,$$

where $\overline{\overline{F}}_c$ stands for the convective and $\overline{\overline{F}}_v$ for the viscous flux tensor. They are in turn given by

$$\overline{\overline{F}}_c = \begin{bmatrix} \rho\, v_x & \rho\, v_y & \rho\, v_z \\ \rho\, v_x^2 + p & \rho\, v_x\, v_y & \rho\, v_x\, v_z \\ \rho\, v_y\, v_x & \rho\, v_y^2 + p & \rho\, v_y\, v_z \\ \rho\, v_z\, v_x & \rho\, v_z\, v_y & \rho\, v_z^2 + p \\ \rho\, H v_x & \rho\, H v_y & \rho\, H v_z \end{bmatrix}$$

and

$$\overline{\overline{F}}_v = \begin{bmatrix} 0 & 0 & 0 \\ \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \\ \kappa_x & \kappa_y & \kappa_z \end{bmatrix}$$

with

$$\kappa_x = v_x\, \tau_{xx} + v_y\, \tau_{xy} + v_z\, \tau_{xz} + k\, \frac{\partial T}{\partial x}, \tag{3.20}$$

$$\kappa_y = v_x\, \tau_{yx} + v_y\, \tau_{yy} + v_z\, \tau_{yz} + k\, \frac{\partial T}{\partial y}, \tag{3.21}$$

$$\kappa_z = v_x\, \tau_{zx} + v_y\, \tau_{zy} + v_z\, \tau_{zz} + k\, \frac{\partial T}{\partial y}. \tag{3.22}$$

Note that the integral of the flux tensor with respect to the surface is called the *flux*.

Finally, the source term $\boldsymbol{Q}$ consists of all integrands of the right hand side of the above equations and reads

$$\boldsymbol{Q} = \begin{bmatrix} 0 \\ f_{bx} \\ f_{by} \\ f_{bz} \\ \rho\, (\boldsymbol{f_b}^T \cdot \boldsymbol{v}) + \dot{q}_h \end{bmatrix}.$$

With these definitions the governing equations can be simplified to

$$\int\limits_{\Omega} \frac{\partial \boldsymbol{W}}{\partial t}\, \mathrm{d}\Omega + \oint\limits_{S} \overline{\overline{F}} \cdot \boldsymbol{n}\, \mathrm{d}S = \int\limits_{\Omega} \boldsymbol{Q}\, \mathrm{d}\Omega. \tag{3.23}$$

The first term on the left hand side of equation (3.23) represents the temporal rate of change of the conservative variables, while the second one symbolizes the flux through the surface of an arbitrary control volume Ω. If the source term is zero[3], equation (3.23) states that the change of the conservative variables and the flux through the surface are in balance. For example, an alteration of energy inside a volume is caused by energy entering or leaving the volume.

If viscous forces are neglected, which essentially means that the viscous flux tensor $\overline{\overline{F}}_v$ is zero, the governing equations are called *Euler equations*. Otherwise they are called *Navier-Stokes equations*.

---

[3] In other words, the body forces and the heat radiation are neglected.

Another distinction for the governing equations is whether the system is changing or unchanging in time. In the former case the equations are named *unsteady* and in the latter case *steady*. Unsteady phenomena include, for example, the occurrence of a wind gust in the flight of an airplane. Steady flow, on the other hand, occurs when the plane has reached cruising altitude and is undisturbed by sudden wind phenomena. In this case, parameters such as the angle of attack and the velocity of the airplane, given by the Mach number, determine the flow.

Note that in the steady equations the first term of equation (3.23), which is the time derivative of the conservative variables, is zero when the solution is found. This solution is called *steady state*, which is the state that is unchanging in time.

In the results presented in Chapter 10 of this thesis, steady states are computed with the help of a reduced order model. Also the snapshots (see Section 1.1) for constructing the reduced order model are steady states. These vary in the angle of attack α and for one test case additionally in the Mach number M_∞. Both α and M_∞ are parameters to the governing equations. In fact, the boundary conditions are dependent on these parameters, as it will be seen in Section 4.4.

3.6 Transition from integral to differential form

This section aims at transforming the governing equations from integral to differential formulation. For this purpose the Gauß theorem is needed, which is presented subsequently.

3.6.1 The Gauß Theorem

The Gauß Theorem is also known as divergence theorem due to the occurrence of *divergence* in one of the integrals. The *divergence* of a vector field $\boldsymbol{f} \in \mathbb{R}^n$ is defined by

$$\mathrm{div}\boldsymbol{f} = \sum_{i=1}^{n} \frac{\partial f_i}{\partial x_i}$$

for the Euclidean vector space $\mathbb{R}^n$.

Theorem 3.1 (Gauß Theorem) *Let $\Omega \subset \mathbb{R}^n$ be compact with a piecewise smooth boundary S. Furthermore, let $\boldsymbol{n}$ be the outer facing normal vector. If $\boldsymbol{f}$ is a continuously differentiable vector field defined on a neighborhood of Ω, then*

$$\int_\Omega \mathrm{div}\boldsymbol{f}\,\mathrm{d}\Omega = \oint_S \boldsymbol{f}^T \cdot \boldsymbol{n}\,\mathrm{d}S. \tag{3.24}$$

Proof: Refer to [33] for the proof and for specifics on the conditions of the theorem. ∎

3.6.2 Application of the Gauß Theorem to the governing equations

Consider the governing equations in scalar form with the source term being zero

$$
\begin{array}{lll}
(1) & \int\limits_\Omega \frac{\partial \rho}{\partial t}\,\mathrm{d}\Omega + \oint\limits_S \rho\,(\boldsymbol{v}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S & = 0,\\
(2) & \int\limits_\Omega \frac{\partial(\rho\, v_x)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_S \rho\, v_x\,(\boldsymbol{v}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S + \oint\limits_S p\,(\boldsymbol{e}_x{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S - \oint\limits_S (\boldsymbol{\tau}_x{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S & = 0,\\
(3) & \int\limits_\Omega \frac{\partial(\rho\, v_y)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_S \rho\, v_y\,(\boldsymbol{v}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S + \oint\limits_S p\,(\boldsymbol{e}_y{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S - \oint\limits_S (\boldsymbol{\tau}_y{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S & = 0,\\
(4) & \int\limits_\Omega \frac{\partial(\rho\, v_z)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_S \rho\, v_z\,(\boldsymbol{v}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S + \oint\limits_S p\,(\boldsymbol{e}_z{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S - \oint\limits_S (\boldsymbol{\tau}_z{}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S & = 0,\\
(5) & \int\limits_\Omega \frac{\partial(\rho\, E)}{\partial t}\,\mathrm{d}\Omega + \oint\limits_S \rho\, H\,(\boldsymbol{v}^T\!\cdot \boldsymbol{n})\,\mathrm{d}S - \oint\limits_S \boldsymbol{v}^T\!\cdot \left(\overline{\overline{\tau}}\cdot \boldsymbol{n}\right)\,\mathrm{d}S + \oint\limits_S \boldsymbol{K}^T\!\cdot \boldsymbol{n}\,\mathrm{d}S & = 0,
\end{array}
$$

where $\boldsymbol{e}_x$, $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$ are the unit vectors and v_x, v_y and v_z are the velocities in the direction specified by the index. Furthermore $\boldsymbol{\tau}_x{}^T = \begin{bmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \end{bmatrix}$ is the first row of the stress tensor $\overline{\overline{\tau}}$. Accordingly, $\boldsymbol{\tau}_y{}^T$ and $\boldsymbol{\tau}_z{}^T$ are the second and third row of $\overline{\overline{\tau}}$.

For the derivation of a differential form of the governing equation Theorem 3.1 will be applied[4] to all surface integrals. This will be exemplified in detail for the mass conservation equation (1) in the above set of equations, which then becomes

$$
\begin{aligned}
0 &= \int\limits_\Omega \frac{\partial}{\partial t}\,\rho\,\mathrm{d}\Omega + \int\limits_\Omega \mathrm{div}(\rho\,\boldsymbol{v})\,\mathrm{d}\Omega \\
&= \int\limits_\Omega \left[\frac{\partial}{\partial t}\,\rho + \mathrm{div}(\rho\,\boldsymbol{v})\right]\mathrm{d}\Omega. \qquad (3.25)
\end{aligned}
$$

Since the integral in equation (3.25) is zero for an arbitrary volume Ω, the integrand has to be zero[5] , i.e.

$$
\frac{\partial}{\partial t}\,\rho + \mathrm{div}(\rho\,\boldsymbol{v}) = \frac{\partial}{\partial t}\,\rho + \frac{\partial(\rho\, v_x)}{\partial x} + \frac{\partial(\rho\, v_y)}{\partial y} + \frac{\partial(\rho\, v_z)}{\partial z} = 0. \qquad (3.26)
$$

With analogous arguments the set of governing equations becomes

$$
\begin{array}{lccccccccc}
(1) & \frac{\partial \rho}{\partial t} & + & \frac{\partial(\rho\, v_x)}{\partial x} & + & \frac{\partial(\rho\, v_y)}{\partial y} & + & \frac{\partial(\rho\, v_z)}{\partial z} & = & 0,\\
(2) & \frac{\partial(\rho\, v_x)}{\partial t} & + & \frac{\partial(\rho\, v_x^2 + p - \tau_{xx})}{\partial x} & + & \frac{\partial(\rho\, v_x\, v_y - \tau_{xy})}{\partial y} & + & \frac{\partial(\rho\, v_x\, v_z - \tau_{xz})}{\partial z} & = & 0,\\
(3) & \frac{\partial(\rho\, v_y)}{\partial t} & + & \frac{\partial(\rho\, v_x\, v_y - \tau_{yx})}{\partial x} & + & \frac{\partial(\rho\, v_y^2 + p - \tau_{yy})}{\partial y} & + & \frac{\partial(\rho\, v_y\, v_z - \tau_{yz})}{\partial z} & = & 0,\\
(4) & \frac{\partial(\rho\, v_z)}{\partial t} & + & \frac{\partial(\rho\, v_x\, v_z - \tau_{zx})}{\partial x} & + & \frac{\partial(\rho\, v_y\, v_z - \tau_{zy})}{\partial y} & + & \frac{\partial(\rho\, v_z^2 + p - \tau_{zz})}{\partial z} & = & 0,\\
(5) & \frac{\partial(\rho\, E)}{\partial t} & + & \frac{\partial(\rho\, H\, v_x - \kappa_x)}{\partial x} & + & \frac{\partial(\rho\, H\, v_y - \kappa_y)}{\partial y} & + & \frac{\partial(\rho\, H\, v_z - \kappa_z)}{\partial z} & = & 0,
\end{array}
\qquad (3.27)
$$

where κ_x, κ_y and κ_z are defined as in equation (3.21), (3.22) and (3.22), respectively.

As it has been done for the integral formulation of the governing equation, the differential

---

[4] The necessary smoothness conditions are assumed.

[5] This can easily be proved by contradiction: Assume the integrand is continuous and nonzero at some point. Then for a volume in the neighborhood of that point the integral is also nonzero.

form shall be expressed in a shorter way. For this purpose the *convective flux vectors*

$$\boldsymbol{F_x}_c = \begin{bmatrix} \rho\, v_x \\ \rho\, v_x^2 + p \\ \rho\, v_x\, v_y \\ \rho\, v_x\, v_z \\ \rho\, H\, v_x \end{bmatrix}, \qquad \boldsymbol{F_y}_c = \begin{bmatrix} \rho\, v_y \\ \rho\, v_x\, v_y \\ \rho\, v_y^2 + p \\ \rho\, v_y\, v_z \\ \rho\, H\, v_y \end{bmatrix}, \qquad \boldsymbol{F_z}_c = \begin{bmatrix} \rho\, v_x \\ \rho\, v_x\, v_z \\ \rho\, v_y\, v_z \\ \rho\, v_z^2 + p \\ \rho\, H\, v_z \end{bmatrix}$$

and the *viscid flux vectors*

$$\boldsymbol{F_x}_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ \kappa_x \end{bmatrix}, \qquad \boldsymbol{F_y}_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ \kappa_y \end{bmatrix}, \qquad \boldsymbol{F_z}_v = \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ \kappa_z \end{bmatrix}$$

are defined. With the definition of

$$\boldsymbol{F_x} = \boldsymbol{F_x}_c - \boldsymbol{F_x}_v, \qquad \boldsymbol{F_y} = \boldsymbol{F_y}_c - \boldsymbol{F_y}_v, \qquad \boldsymbol{F_z} = \boldsymbol{F_z}_c - \boldsymbol{F_z}_v$$

and the vector of conservative variables $\boldsymbol{W}$ defined in equation (3.19) the differential form becomes

$$\frac{\partial \boldsymbol{W}}{\partial t} + \frac{\partial \boldsymbol{F_x}}{\partial x} + \frac{\partial \boldsymbol{F_y}}{\partial y} + \frac{\partial \boldsymbol{F_z}}{\partial z} = 0. \tag{3.28}$$

This equation (3.28) is called the *strong conservation form* of the governing equations.

Up to now, the source term has been neglected. Note that it is the integrand of a volume integral. Therefore the Gauß Theorem is not applied to it and it remains unchanged. As a result the strong conservation form with consideration of the source term is

$$\frac{\partial \boldsymbol{W}}{\partial t} + \frac{\partial \boldsymbol{F_x}}{\partial x} + \frac{\partial \boldsymbol{F_y}}{\partial y} + \frac{\partial \boldsymbol{F_z}}{\partial z} = \boldsymbol{Q}. \tag{3.29}$$

3.7 Turbulent flows and the Reynolds-averaged Navier-Stokes Equations

The governing equations that have been derived in the previous sections can be solved numerically. However, due to the small-scale fluctuations of turbulent flows the control volumes have to be very small in order to resolve these fluctuations. This leads to enormous computational complexity and makes relevant aerodynamic problems in aerospace infeasible.[6]

A way out of this are the so-called *Reynolds-averaged Navier-Stokes (RANS) equations*. The idea is to model the turbulence by *turbulence models* instead of resolving them [35, p. 32].

---

[6] The direct simulation of turbulence by the Navier-Stokes equations is called *Direct Numerical Simulation (DNS)* [10, p.225].

In order to derive the RANS equations the flow unknowns such as v_x, v_y and v_z, are expressed as the sum of a low-frequency average $\bar{v}_x$, $\bar{v}_y$ and $\bar{v}_z$ and a superimposed high-frequency fluctuation term $\tilde{v}_x$, $\tilde{v}_y$ and $\tilde{v}_z$:

$$\begin{aligned} v_x &= \bar{v}_x + \tilde{v}_x \\ v_y &= \bar{v}_y + \tilde{v}_y \\ v_z &= \bar{v}_z + \tilde{v}_z \end{aligned}$$

This leads to a similar set of equations as the original equations, with the difference that the flow unknowns v_x, v_y and v_z in the equations are replaced by their average and furthermore additional stresses, called *Reynolds stresses*, occur. These stresses represent the turbulence and are substituted by so-called *turbulence models*. Since the turbulence models handle the small-scale fluctuations, the governing equations do not have to resolve them anymore. Thus, less resolution in time and space is required, which reduces the computational complexity significantly [18, 35].

3.7.1 First-order closures

For this thesis only the so-called first-order closure turbulence models are of interest. Other turbulence model like second-order closures (Reynolds-Stress models) will not be used in this work. Although turbulence models are important for obtaining accurate CFD solutions, a detailed description of the employed turbulence models goes beyond the scope of this thesis. However, since they are part of the overall RANS system, the first-order closures shall be sketched briefly in the following. The interested reader is referred to [10, Chapter 7] and the references therein.

First-order closures are based on the Boussinesq eddy-viscosity hypothesis, which states that the turbulent shear stress is linearly related to the mean rate of strain [10, p.223], where the *eddy viscosity* μ_T acts as a proportionality factor.

Representatives of first-order closures are the Spalart-Allmaras one-equation as well as the K-ϵ and K-ω SST (Shear-Stress Transport) two-equation models. They all try to approximate the eddy viscosity μ_T with the help of different variables. The former uses a single, eddy-viscosity type of variable $\tilde{\nu}$, while the latter ones use the turbulent kinetic energy K and the turbulent dissipation rate ϵ or the specific dissipation of turbulence ω. In integral form the different turbulence models can be formally written as

$$\int_\Omega \frac{\partial \boldsymbol{W}_T}{\partial t}\,\mathrm{d}\Omega + \oint_S \overline{\overline{F}}_T \cdot \boldsymbol{n}\,\mathrm{d}S = \int_\Omega \boldsymbol{Q}_T\,\mathrm{d}\Omega, \tag{3.30}$$

where $\boldsymbol{W}_T = \tilde{\nu}$, $\boldsymbol{W}_T^T = [\rho K \ \rho\epsilon]$ or $\boldsymbol{W}_T^T = [\rho K \ \rho\omega]$ depending on the turbulence model. For details on the turbulent flux tensor $\overline{\overline{F}}_T$ and the source term $\boldsymbol{Q}_T$ the reader is referred to [10].

Note that equation (3.30) is appended to the governing equations (3.23) in case of the Reynolds-averaged Navier-Stokes (RANS) equations.

Chapter 4

Computational fluid dynamics

In the previous chapter the Euler and Navier-Stokes equations have been derived. Since only few analytical solutions exist for very simple flow problems, the governing equations are mainly solved numerically. This is done in three steps: At first a *computational grid* (also called *mesh*) is generated. Then the governing equations are discretized in space according to the defined grid which leads to a system of *ordinary differential equations* (ODE). Finally, the ODE system is discretely advanced in time to yield the solution. In the following an overview of the three steps is given. As a main source for this section the textbook by Blazek [10] is used.

4.1 Grid generation

The first step in solving the governing equations numerically is to discretize the *physical space*. This is done by dividing the space into small geometric elements, called *grid cells*. For two-dimensional spaces the grid cells are usually triangles or quadrilaterals (see Figure 4.1) and in three-dimensional spaces they are tetrahedra, hexahedra, prisms, or pyramids (see Figure 4.2).

When the grid is generated, two obvious properties have to be fulfilled: Firstly, there must not be any gaps in between the grid cells and secondly, there must not be an overlap between cells. In order to achieve good accuracy the grid should additionally be *smooth*, that is, neighboring grid cells should be of about the same size and form.

An important distinction between different grids is made in the structure of the grid cells. A grid is called *structured*, if the position of the grid points, which are the vertices of the grid

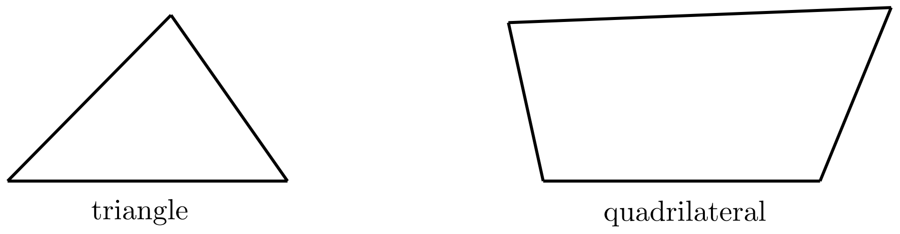
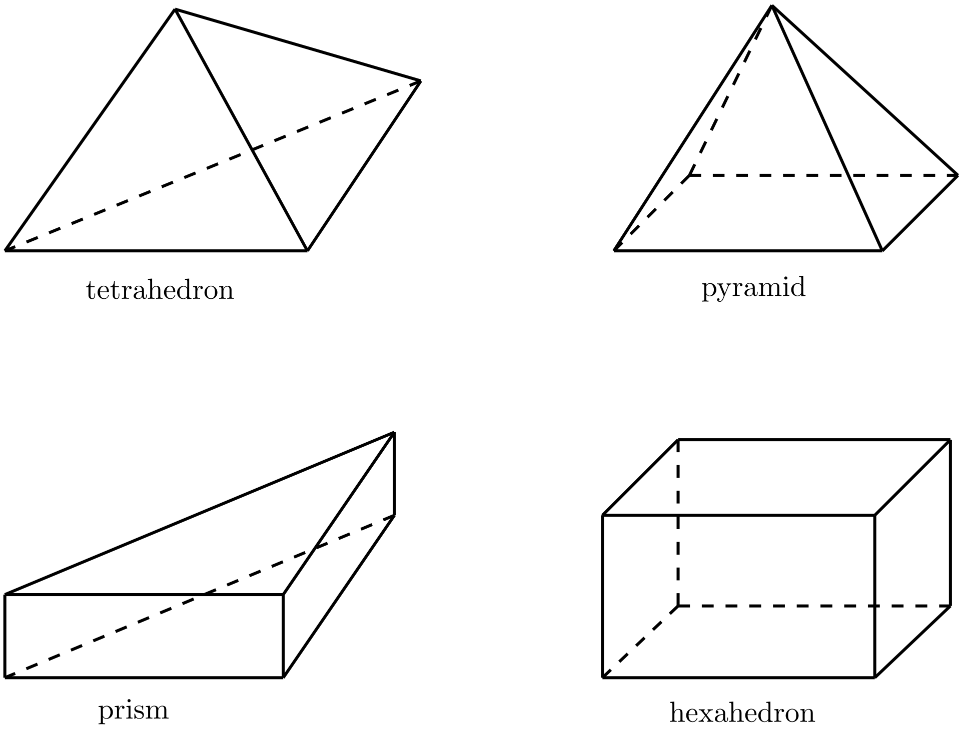


Figure 4.1: *The shapes of grid cells in two dimensions.*



tetrahedron pyramid

prism hexahedron

Figure 4.2: *The various shapes of grid cells of three-dimensional unstructured grids.*

cells, can be identified with the help of a set of indices (i, j, k). That is, if one of the indices is decremented or incremented a neighboring grid point is found in x-, y-, or z-direction. This is illustrated in Figure 4.3(a). In two-dimensional space the grid cells are quadrilaterals and in three-dimensional space they are hexahedra.

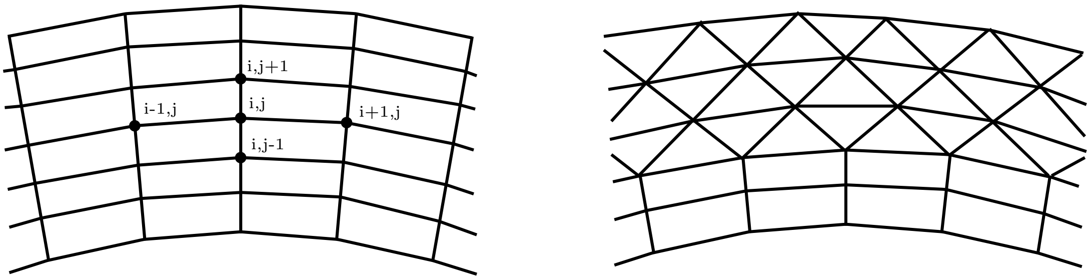
Unstructured grids on the other hand are more flexible in the treatment of complex geometries than structured grids. Their grid points are not ordered and a variety of different shapes for the grid cells can be chosen. In two dimensions usually triangles are used; sometimes in combination with quadrilaterals. In three dimensions tetrahedra are most common, but some flow solvers also employ a mixture of tetrahedra, prisms, pyramids, and occasionally also hexahedra (see Figure 4.2).

In order to resolve the boundary layers accurately, often *hybrid grids* are employed, which are structured near the solid body and unstructured further away. Such a hybrid grid is illustrated in Figure 4.3(b).

Although grid generation is a crucial ingredient to obtaining accurate solutions, it shall not be tackled in detail in this thesis, since the focus is on model order reduction of the governing equations of fluid mechanics.

4.2 Spatial discretization

In the following the wide-spread finite volume discretization will be depicted. Other discretization techniques such as finite difference and finite element will not be investigated, but



(a) A structured grid.

(b) A hybrid grid.

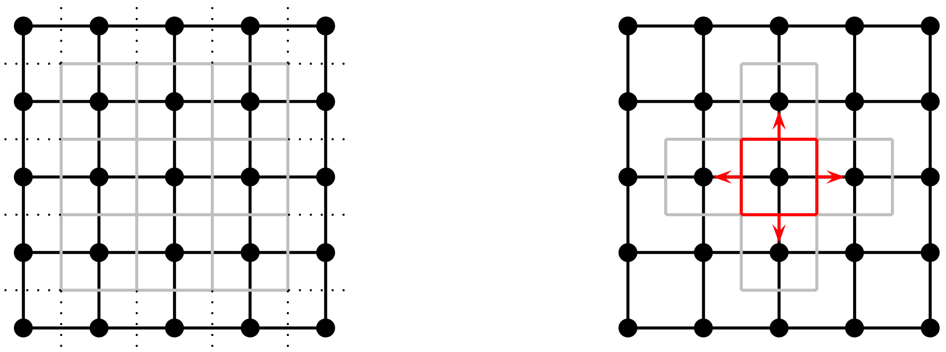
Figure 4.3: *An illustration of a structured and a hybrid grid.*

can be found in the literature such as [10] and the references therein.

In the finite volume discretization, after the physical domain has been subdivided into grid cells, the governing equations are discretized for each grid cell. Different methods of how to define the control volumes with respect to the grid are available, among which the *cell-centered scheme* and the *cell-vertex scheme* with dual control volumes are the most prominent.

In the *cell-centered* scheme the control volumes and the grid cells are identical. It is assumed that the flow variables are constant in the control volumes and they are thus associated with the centroids.

In the *cell-vertex* scheme with dual control volumes a dual grid is created. In the case of structured grids this is achieved by connecting the midpoints of neighboring cells that have a vertex in common (see Figure 4.4(a)). For unstructured grids the dual grid is constructed by connecting the midpoints of surrounding cells, face-centroids and edge-midpoints. Consequently, in both cases the so-called *grid points*, which are the vertices of the computational grid, are surrounded by the cells of the dual grid. Note that the control volumes are identical to the cells of the dual grid. The flow variables are once again constant in a volume and are located at the grid points. In this thesis the CFD computations will employ the dual grid approach.



(a) A dual grid (gray lines) and the underlying (primary) grid.

(b) The faces (red lines) and the fluxes (red arrows) of a control volume.

Figure 4.4: *Illustration of the cell-vertex scheme with dual control volumes*

For simplicity, assume that the grid and thus also the control volumes are not changing with time. In this case the time derivative of the conservative variables can be written as

$$\frac{\partial}{\partial t}\int\limits_{\Omega} \boldsymbol{W}\,\mathrm{d}\Omega = \Omega\frac{\partial \boldsymbol{W}}{\partial t}.$$

The governing equations (3.23) thus become

$$\frac{\partial \boldsymbol{W}}{\partial t} = -\Omega^{-1}\left[\oint\limits_{S} \overline{\overline{F}}\cdot\boldsymbol{n}\,\mathrm{d}S - \int\limits_{\Omega}\boldsymbol{Q}\,\mathrm{d}\Omega\right]. \tag{4.1}$$

The discretization of the finite volume method lies in the evaluation of the two integrals in the above equation based on the flow variables at discrete points. The approximation of the surface integral is achieved by a sum of fluxes passing through the faces of the control volume. Each flux is approximated by assuming that it is constant throughout the face and that it is located at the midpoint of the face (see Figure 4.4(b)). The source term is often also considered to be constant within the control volume. For a control volume Ω_i, where i denotes the index corresponding to a grid point within the dual control volume, equation(4.1) thus becomes

$$\frac{\mathrm{d}\boldsymbol{W}_i}{\mathrm{d}t} = -\Omega_i^{-1}\left[\sum_{j=1}^{n_{f,i}}\left(\overline{\overline{F}}_c - \overline{\overline{F}}_v\right)_j \Delta S_j - (\boldsymbol{Q}\Omega)_i\right]. \tag{4.2}$$

In the above equation ΔS_j stands for the area of the face j and $n_{f,i}$ for the number of faces of the control volume Ω_i. The number of faces $n_{f,i}$ depends on the control volume and on the kind of grid being used for unstructured grids. For structured ones often $n_{f,i} = 4$ in the two-dimensional space and $n_{f,i} = 6$ in three spatial dimensions.

The quality of the discretization is highly dependent on the approximation of the convective and viscous fluxes. The key issue is that they have to be known at the midpoint of all $n_{f,i}$ faces of a control volume Ω_i, but the flow variables are not directly accessible there, since they are stored at locations within the control volumes (see Figure 4.4(b)). Consequently, either the fluxes or the flow variables, which are needed for the evaluation of the fluxes, have to be interpolated to these points.

For the convective fluxes the interpolation is based on one of the following ideas: In *central* discretization schemes arithmetic averaging is used, while in *upwind* schemes the interpolation is biased depending on the wave propagation directions.

Since upwind schemes take the flow physics more accurately into account, they are able to capture discontinuities a lot better than central schemes. A wide variety of upwind schemes have been developed. They can be divided roughly into the four groups: *flux-vector splitting*, *flux-difference splitting*, *total variation diminishing*, and *fluctuation splitting* schemes. Please refer to [10] for a more detailed account of the spatial discretization methods.

An advantage of central schemes is, generally speaking, that they are computationally more efficient. They make use of some kind of numerical dissipation in order to ensure stability. Again, the readers are referred to [10] for details.

For the viscous fluxes arithmetic averaging is very often employed.

4.3 Temporal discretization

Most numerical schemes use the *method of lines*, that is, the governing equations are separately discretized for space and time. After the spatial discretization the governing equations (3.23) become a system of ordinary differential equations of the form (4.2). The term in square brackets in (4.2) is named *residual* and is a non-linear function of the vector of conservative variables[1] $\boldsymbol{w}(t)$. It is abbreviated by $\boldsymbol{R}$, which motivates the short formulation

$$\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = -\Omega^{-1}\boldsymbol{R}(\boldsymbol{w}(t)). \tag{4.3}$$

Note that in case the grid points do not coincide with the centroids of the control volumes, a mass matrix M has to be introduced on the left hand side, in order to compute a time-accurate solution [66, p. 22]. The left hand side then becomes $\frac{\mathrm{d}(M\boldsymbol{w}(t))}{\mathrm{d}t}$. With the help of the mass matrix M the average value of the conservative variables in the control volume is computed with the values at the grid point associated with the control volume and its neighboring grid points. In this way, the mass matrix M couples the system of ordinary differential equations.

Note that the average value of the conservative variables in the control volume is well approximated by the value in the centroid [66, p. 22]. Since for cell-centered schemes the grid points correspond to the centroids, the mass matrix M can be replaced by the identity matrix. For cell-vertex schemes, the grid points generally do not coincide with the cell centroids. Consequently, $M \neq I$.

For steady state computations the time derivative, in which the mass matrix occurs, is zero. Thus, for these computations M can be lumped [66]. That is, it can be replaced by the identity matrix.

There are two different schemes to advance the equations (4.3) in time: *Explicit schemes* make use of only those residuals, which are known in the current time step, while *implicit schemes* also utilize unknown residuals at the current time step.

Both schemes have advantages and disadvantages. While implicit schemes have the quality that the time step can be chosen significantly larger without losing stability and are generally more robust, explicit schemes offer a higher computational efficiency per time step and are easier to parallelize.

In the following subsections both schemes will be described. Note that although the time derivative drops out in the steady equations, time-stepping approaches are used. In fact, to solve the steady equations – starting from an initial guess – the system is advanced in time until the change in the solution becomes insignificant.

4.3.1 Explicit schemes

In explicit schemes the solution $\boldsymbol{w}$ at the next time step $t_{n+1} = t_n + \Delta t$ is computed with the help of the knowledge of the solution and the residual at the current time step t_n and possibly

---

[1] Note that we will denote the discrete vector of conservative variables with $\boldsymbol{w}(t)$, while the one containing functions for the density, velocities and pressure is written as $\boldsymbol{W}(t)$.

also other previous time steps. From an implementational point of view, this means that already known vectors have to be added up. This makes explicit methods very popular. However, it has to be noted that explicit schemes are not as stable as implicit ones, which results in the usage of smaller time steps.

In most CFD codes the so-called *multi-stage* scheme is used. Multi-stage schemes advance the solution in a number of stages, say m stages, where each stage can be seen as a sequence of updates according to (4.3). The following formulae characterize the scheme

$$\begin{aligned}
\boldsymbol{w}^{(0)} &= \boldsymbol{w}(t_n) \\
\boldsymbol{w}^{(1)} &= \boldsymbol{w}^{(0)} - \alpha_1 \Delta t \, \Omega^{-1} \cdot \boldsymbol{R}\left(\boldsymbol{w}^{(0)}\right) \\
\boldsymbol{w}^{(2)} &= \boldsymbol{w}^{(0)} - \alpha_2 \Delta t \, \Omega^{-1} \cdot \boldsymbol{R}\left(\boldsymbol{w}^{(1)}\right) \\
&= \ldots \\
\boldsymbol{w}^{(m)} &= \boldsymbol{w}^{(0)} - \alpha_m \Delta t \, \Omega^{-1} \cdot \boldsymbol{R}\left(\boldsymbol{w}^{(m-1)}\right) = \boldsymbol{w}(t_{n+1}).
\end{aligned} \tag{4.4}$$

The specific choice of the stage coefficients α_k are not of importance for this thesis, but they can be found in [10, p.183]. Note that the mass matrix M has been lumped (i.e., replaced by the identity matrix) for simplicity. For time-accurate computations it has to be included.

The time step Δt has to be chosen such that the above scheme remains stable. For this reason the *Courant-Friedrichs-Lewy (CFL)* condition[2] has to be satisfied, which states that the domain of dependence of the numerical method has to contain at least the domain of dependence of the partial differential equations. This means that the propagation speed of information in the grid has to be smaller than the physical propagation speed of a disturbance such as an acoustic wave [35, p. 59]. How this is done in practice can be found in [10, p. 186].

For the computation of steady states different acceleration techniques for speeding up the convergence are available. While for time-accurate computations the time step has to be chosen such that it is the smallest locally permissible step, this is not the case for steady state computations. Instead, for each grid point, the maximal locally permissible time step is chosen. This approach is called *local time-stepping*. Obviously, intermediate solutions do not correspond to a physical time, but the final converged solution, which does not change in time, is the sought after steady state.

Other acceleration techniques are *residual smoothing* and most importantly the *multigrid* approach. Details can be found in Chapter 9 of [10] and the references therein.

4.3.2 Implicit schemes

Unlike explicit schemes, implicit ones make use of residuals evaluated at the next time step $t_{n+1} = t_n + \Delta t$, that is, $\boldsymbol{R}\left(\boldsymbol{w}(t_{n+1})\right)$. A general class of implicit schemes is given by

$$\frac{(M\Omega)}{\Delta t} \, \Delta \boldsymbol{w}(t_n) = -\beta \cdot \boldsymbol{R}(\boldsymbol{w}(t_{n+1})) - (1-\beta) \cdot \boldsymbol{R}(\boldsymbol{w}(t_n)), \tag{4.5}$$

where $\Delta \boldsymbol{w}(t_n) = \boldsymbol{w}(t_{n+1}) - \boldsymbol{w}(t_n)$ and β is a parameter, which can be chosen. Since the conservative variables for the next time step $\boldsymbol{w}(t_{n+1})$ are unknown, its residual $\boldsymbol{R}\left(\boldsymbol{w}(t_{n+1})\right)$ can

---

[2] This is due to the hyperbolic nature of the governing equations with respect to time.

not be evaluated directly. An approach for overcoming this obstacle is given by an approximate Taylor series expansion

$$\boldsymbol{R}\left(\boldsymbol{w}(t_{n+1})\right) \approx \boldsymbol{R}(\boldsymbol{w}(t_n)) + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}} \cdot \Delta \boldsymbol{w}(t_n). \tag{4.6}$$

This is injected into the implicit scheme (4.5), which yields

$$\left[\frac{(M\Omega)}{\Delta t} + \beta \cdot \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}\right] \Delta \boldsymbol{w}(t_n) = -\boldsymbol{R}(\boldsymbol{w}(t_n)). \tag{4.7}$$

As it has been stated before, temporal accuracy is not an issue in case of steady state computations. This is the reason why the mass matrix can be neglected in (4.7). For the same reason the parameter β is preferably set to 1, in which case the temporal discretization is more robust, but of lower order – which means less accurate – than if $\beta = 1/2$.

The term in square brackets on the left hand side in (4.7) is called the *implicit operator* and stands for a large sparse matrix. There are two different possibilities to solve the system (4.7): Either direct or iterative methods can be employed.

Direct methods compute an exact inverse of the implicit operator. This is achieved with the help of LU decomposition or direct sparse methods. These direct methods, however, are not advisable for large three-dimensional problems, since they require high computational effort, if they are feasible at all.

The alternative are iterative methods, which can be divided into two different types: The first type factorizes the implicit operator into factors, which can be more easily inverted. Among such methods are the *Alternating Direction Implicit (ADI)*, which can only be used for structured grids, the *Jacobi* or the *Gauss-Seidel relaxation* and the *Lower-Upper Symmetric Gauss-Seidel (LU-SGS)* scheme [10, Section 6.2].

The second type of iterative methods use Krylov-subspace methods for the inversion of the implicit operator. Such methods are the *Conjugate Gradient Squared (CGS)*, the *Bi-Conjugate Gradient Stabilized (Bi-CGSTAB)*, the *Transpose-Free Quasi-Minimum Residual (TFQMR)*, and most importantly the *Generalized Minimal Residual (GMRES)* method [10].

Usually, for steady state computations the term $\frac{(M\Omega)}{\Delta t}$ is dropped by setting $\Delta t \to \infty$. Equation (4.7) thus becomes

$$\beta \cdot \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}} \cdot \Delta \boldsymbol{w}(t_n) = -\boldsymbol{R}(\boldsymbol{w}(t_n)), \tag{4.8}$$

which is in fact Newton's method [56, 42, 14] for finding the root of the residual $\boldsymbol{R}$, if β is chosen to be one. Obviously, the root of the residual $\boldsymbol{R}$ is the steady state.

Due to the importance of Newton's method, it shall be outlined briefly in the following: It is an iterative approach, which starting from an initial guess seeks better approximants of the root in each iteration step. Obviously, the goal is to find $\boldsymbol{w}(t_{n+1})$ such that $\boldsymbol{R}\left(\boldsymbol{w}(t_{n+1})\right) = 0$. By inserting this goal into the approximate Taylor expansion (4.6) Newton's method (4.8) is obtained, where $\beta = 1$ in (4.8). Note that since an *approximate* Taylor expansion is used, (4.8) will not immediately yield the exact solution, but an approximation of $\boldsymbol{w}(t_{n+1}) = \boldsymbol{w}(t_n) + \Delta \boldsymbol{w}(t_n)$. Hence, this process has to be repeated until the method has sufficiently converged. In fact, it can be shown that under certain conditions Newton's method converges quadratically, that is informally speaking, the number of correct digits nearly double in each iteration [14].

Note, however, that problems associated with Newton's method are its high memory requirements and the choice of a good start solution.

4.4 Initial and boundary conditions

In order to be able to solve the governing equations numerically *initial* and *boundary conditions* have to be specified.

Initial conditions: Initial conditions describe the state of the flow at the first time step. Obviously, the closer the initial condition is to the sought solution, the faster the method will converge and the more robust it will be. In the CFD software TAU [22, 21] of the German Aerospace Center as well as in many other CFD solvers, the default initial condition is given by freestream values of pressure, density, and velocity components in the entire flow field. This means that at each grid point the values are set to a reference value.

In case of the Euler equations the default values for pressure and temperature in the TAU code [28, p. 71] are $\bar{p}_{ref} = 101325.0\frac{N}{m^2}$ and $\bar{T}_{ref} = 273.15K$, respectively, which describe air at normal atmospheric conditions. These values can be overwritten to fulfill the given needs. Density is then given by $\bar{\rho}_{ref} = \frac{\bar{p}_{ref}}{\bar{T}_{ref} \cdot R}$, where (3.15) is used and R is the gas constant.

For viscous flows modeled by the Navier-Stokes equations often the speed of the aerodynamic body (the Mach number), the temperature and the Reynolds number are given (e.g. from a wind tunnel experiment). The density can then be computed with the help of the definition of the Reynolds number (2.1) and the Sutherland formula (3.18). Subsequently, the value of pressure is obtained from (3.15).

In both Euler and Navier-Stokes equations the velocity components are finally obtained by

$$\begin{aligned}
\bar{v}_{x,ref} &= \cos(\alpha) \cdot \cos(\beta) \cdot M_\infty \cdot \sqrt{\gamma \frac{\bar{p}_{ref}}{\bar{\rho}_{ref}}} \\
\bar{v}_{y,ref} &= \sin(\beta) \cdot M_\infty \cdot \sqrt{\gamma \frac{\bar{p}_{ref}}{\bar{\rho}_{ref}}} \qquad (4.9) \\
\bar{v}_{z,ref} &= \sin(\alpha) \cdot \cos(\beta) \cdot M_\infty \cdot \sqrt{\gamma \frac{\bar{p}_{ref}}{\bar{\rho}_{ref}}}
\end{aligned}$$

with the freestream Mach number M_∞, the angle of attack α and the sideslip angle β, which are parameters to the governing equations and have to be set.

A second possibility for an initial condition is to use a solution, which is available from previous computations. Ideally, the flow condition of the initial condition is close to the one to be computed. For example, if the steady solution at an angle of attack $\alpha = 2°$ is known, then it can be used to compute the flow at $\alpha = 2.5°$.

Boundary conditions: In external aerodynamics two types of boundary conditions are distinguished: the *farfield* boundary conditions and the boundary conditions at the surface of the body under consideration.

Since a numerical method can simulate the flow only in a finite subdomain of the physical domain, artificial boundaries are created where the computational domain ends. These are the farfield boundary conditions which on the one hand should not be too close to the surface of the body so that the solution yields accurate results. On the other hand they should not be too far from the surface in order to reduce computational complexity. The treatment of the boundaries depends on the speed of the flow, i.e. whether it is sub- or supersonic, and on the direction of the flow, that is, whether it is entering or leaving the computational domain. The direction of the flow will be referred to as *in-* or *outflow*.

The consideration of the speed of the flow has to do with the way that information is propagated in the governing equations, which is determined with the help of characteristics. In fact, the number of characteristics which come into the computational domain should be identical to the number of conditions imposed from outside of the computational domain onto the boundary. All other conditions have to imposed from the inside. Note that the overall number of boundary conditions has to be equal to the number of equations given. In the following the three-dimensional Euler equations are used as a reference, where there are altogether five equations.

Since for supersonic inflow there are five incoming characteristics, all conditions are imposed from the outside of the domain onto the boundary. In fact, the conservative variables at the boundary are determined by the freestream values (p_{ref}, ρ_{ref}, $v_{x,ref}$, $v_{y,ref}$, $v_{z,ref}$) only. In case of supersonic outflow, on the other hand, they are evaluated with the help of the solution from within the computational domain only. This is different for subsonic in- and outflow. Since in case of subsonic inflow there are four incoming characteristics and a single outgoing one, four of the characteristic variables are determined with the help of freestream values. The remaining variable is extrapolated with the help of values of the flow variables from within the computational domain. On the other hand, for subsonic outflow four of the characteristic variables are extrapolated from the interior of the computational domain onto the boundary and the last one is set to a prescribed value [10, 70].

The boundary conditions at the surface of the body depend on the governing equations, which are solved. In case of the Euler equations, which simulate inviscid flow, the boundary conditions state the velocity of the flow normal to the surface is zero, i. e.

$$\boldsymbol{v}^T \boldsymbol{n} = 0.$$

In other words, the flow has to be tangential to the surface and cannot pass through it.

In case of the Navier-Stokes equations it is assumed that the velocity of the flow directly on the surface of the body is zero. This is the so-called *no-slip* boundary condition and reads

$$v_x = v_y = v_z = 0.$$

For the temperature often an adiabatic wall is assumed, which means that there is no heat flux through the wall, i.e.

$$\nabla T \cdot \boldsymbol{n} = \boldsymbol{0}.$$

For more details on initial and boundary conditions please refer to [10].

4.5 Non-dimensionalization of the flow quantities

Note that the flow variables have different scales. For example in Section 4.4 it can be seen that the reference quantity for pressure is $\bar{p}_{ref} = 101325.0\frac{N}{m^2}$, while for density it is $\bar{\rho}_{ref} = 1.2925088\frac{kg}{m^3}$ at a temperature of $\bar{T}_{ref} = 237.15K$. In order to avoid numerical deficiencies due to these different scales, the flow quantities will internally be non-dimensionalized, as in the Tau code [28, p. 71].

This is achieved in case of pressure, density and temperature by dividing the flow variable with its reference quantity. For example the non-dimensional pressure p will be obtained via $p = \frac{\bar{p}}{p_{ref}}$, where $\bar{p}$ is the dimensional pressure.

The Cartesian velocity components on the other hand are non-dimensionalized by scaling with $\sqrt{\frac{\bar{\rho}_{ref}}{\bar{p}_{ref}}}$, e.g.

$$v_x = \bar{v}_x \cdot \sqrt{\frac{\bar{\rho}_{ref}}{\bar{p}_{ref}}}. \tag{4.10}$$

Remember that the velocity, pressure, and density are measured in $\frac{m}{s}$, $\frac{N}{m^2} = \frac{kg}{s^2 \cdot m}$, and $\frac{kg}{m^3}$, respectively. It can easily be seen that the dimensions in (4.10) cancel out.

4.6 Summary

In this chapter it has been shown how the Euler and Navier-Stokes equations are treated numerically. At first, the computational grid is generated. This is done by subdividing the physical space into small grid cells. With the computational grid at hand, the governing equations can be discretized in space. This yields a system of ordinary differential equations given by

$$\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = -\Omega^{-1}\boldsymbol{R}(\boldsymbol{w}(t)). \tag{4.11}$$

Solutions to the system are then obtained by solving it in time, which is called temporal discretization.

These solutions are either specific points in time or steady states. The later are solutions which are unchanging in time. That is, in steady state the residual $\boldsymbol{R}(\boldsymbol{w}(t))$ is (approximately) zero. In this thesis steady state solutions will be the center of attention. They are computed for different flow conditions specified by the angle of attack and Mach numbers. Definitions for these parameters have been given in Section 2.4.

Part II

Model order reduction

Chapter 5

Useful mathematical background for model order reduction

In this chapter two basic mathematical methodologies are regarded. These are the singular value decomposition on the one hand and projections on the other. Both are fundamental for understanding the reduced order methods of this work and play a significant role in model order reduction. The singular value decomposition is related to the proper orthogonal decomposition, which is the method used in this work to create a basis for the reduced subspace. Projections on the other hand are used to reduce the number of governing equations.

Due to their significance, both methodologies are described in detail. Other fundamentals such as concepts from linear algebra like the kernel, range, rank, and column span of a matrix are assumed to be known to the readers. These operations shall be denoted by null(.), range(.), rank(.), and span(.) and their definitions can be found in most standard textbooks, e.g. [54, Chapter 1]. The same holds true for different norms and Hilbert spaces, which shall not be discussed here.

5.1 The singular value decomposition

In this section the *singular value decomposition* (SVD) will be outlined. It is a very useful tool for analyzing matrices. In particular, the SVD is very helpful for determining the (numerical) rank of a matrix and optimal low-rank approximations. The later property will be used to construct a low-dimensional basis for solutions of reduced order models. This section is based on [23] and [1].

First of all, the SVD will be mathematically defined. The following theorem gives a definition and states that the SVD exists for any real matrix.

Theorem 5.1 *Let* $p = \min\{m, n\}$*. If* $A \in \mathbb{R}^{n\times m}$*, then a factorization of the form*

$$A = U\Sigma V^T \tag{5.1}$$

exists, where

$$U = (\boldsymbol{u}_1 \ \dots \ \boldsymbol{u}_n) \in \mathbb{R}^{n\times n}, \qquad V = (\boldsymbol{v}_1 \ \dots \ \boldsymbol{v}_m) \in \mathbb{R}^{m\times m},$$

are orthogonal matrices and the matrix $\Sigma \in \mathbb{R}^{n\times m}$ *is given by* $\Sigma_{ii} = \sigma_i$ *with* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$ *and it is zero elsewhere.*

Proof: See proof of Theorem 2.5.2 in [23]. ∎

The σ_i with $i = 1,\ldots,p$ are called *singular values*. Furthermore the column vectors $\boldsymbol{u}_i$ with $i = 1,\ldots,n$ and $\boldsymbol{v}_i$ with $i = 1,\ldots,m$ of the matrices U and V are referred to as the *left* and *right singular vectors*, respectively. Since orthogonal matrices can be interpreted as rotations or reflections and diagonal matrices as stretchings in Euclidean space, the SVD can be considered as a rotation/reflection, followed by a stretching and another rotation/reflection.

5.1.1 Properties of the SVD

In the following, two important properties with respect to Proper Orthogonal Decomposition will be described.

The SVD and its corresponding eigenvalue decomposition

Due to the orthogonality of the matrices U and V

$$
\begin{aligned}
AA^T &= U\Sigma V^T V\Sigma U^T = U\Sigma^2 U^T, && (5.2)\\
A^T A &= V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T && (5.3)
\end{aligned}
$$

holds. Thus, the left singular vectors $\boldsymbol{u}_j$ with $j \in \{1,...,n\}$ of A are the eigenvectors of AA^T with the eigenvalues σ_j^2, since

$$AA^T\boldsymbol{u}_j = U\Sigma^2U^T\boldsymbol{u}_j = \sigma_j^2\boldsymbol{u}_j.$$

With the same argument it follows that the right singular vectors $\boldsymbol{v}_j$ with $j \in \{1,...,m\}$ are the eigenvectors of A^TA with respect to the eigenvalue σ_j^2.

Low-rank matrix approximation with the SVD

In the following, it will be shown that a matrix $A \in \mathbb{R}^{n\times m}$ can be approximated optimally by a matrix of lower rank with the help of truncating the SVD.

Assume that $B = \tilde{U}\tilde{\Sigma}\tilde{V}^T \in \mathbb{R}^{n\times m}$, where the entries of the singular value matrix $\tilde{\Sigma}$ are given by $\tilde{\sigma}_1 \geq \ldots \geq \tilde{\sigma}_r > \tilde{\sigma}_{r+1} = \cdots = \tilde{\sigma}_p = 0$. The factors of the SVD $\tilde{U}$, $\tilde{V}$, and $\tilde{\Sigma}$ are partitioned into

$$\tilde{U} = \begin{pmatrix} U_r & \bar{U} \end{pmatrix},\ \tilde{V} = \begin{pmatrix} V_r & \bar{V} \end{pmatrix} \text{ and } \tilde{\Sigma} = \begin{bmatrix} \Sigma_r & 0 \\ 0 & \bar{\Sigma} \end{bmatrix}, \tag{5.4}$$

where $U_r \in \mathbb{R}^{n\times r}$ and $V_r \in \mathbb{R}^{m\times r}$ are the first r columns of $\tilde{U}$ and $\tilde{V}$ and $\Sigma_r = \operatorname{diag}(\tilde{\sigma}_1,\ldots,\tilde{\sigma}_r) \in \mathbb{R}^{r\times r}$ and $\bar{\Sigma} = 0 \in \mathbb{R}^{(n-r)\times(m-r)}$.

Since

$$B \cdot V_r = U_r \cdot \Sigma_r$$

and

$$B \cdot \bar{V} = \bar{U} \cdot \bar{\Sigma} = 0,$$

it follows that

$$\begin{aligned} \mathrm{range}(B) &= \mathrm{span}\{\tilde{\boldsymbol{u}}_1, \dots, \tilde{\boldsymbol{u}}_r\}, \\ \mathrm{null}(B) &= \mathrm{span}\{\tilde{\boldsymbol{v}}_{r+1}, \dots, \tilde{\boldsymbol{v}}_m\}, \end{aligned}$$

where $\tilde{\boldsymbol{u}}_j$ for $j = 1, \dots, r$ and $\tilde{\boldsymbol{v}}_j$ for $j = r+1, \dots, m$ are the columns of U_r and $\bar{V}$, respectively. From linear algebra it is well known that dim(range(B)) = rank(B) and this implies that

$$\mathrm{rank}(B) = r.$$

Thus the rank of the matrix B is given by the greatest index, for which the singular value is not zero, that is

$$\mathrm{rank}(B) = \max_j \{j \ : \ \tilde{\sigma}_j \neq 0\}.$$

Note that the SVD of the matrix B of rank r can be shortened to

$$B = U_r \Sigma_r V_r^T = \sum_{i=1}^{r} \tilde{\sigma}_i \tilde{\boldsymbol{u}}_i \tilde{\boldsymbol{v}}_i^T. \tag{5.5}$$

Now assume that the left and right singular vectors for A and B are the same and that $\sigma_i(A) = \tilde{\sigma}_i = \sigma_i(B)$ with $i = 1, \dots, r$. The following theorem states that matrix A can be best approximated by matrix B of the form (5.5).

Theorem 5.2 (Schmidt-Eckart-Young-Mirsky-Theorem) *Let* $A = U\Sigma V^T \in \mathbb{R}^{n\times m}$ *and* $r < k = \mathrm{rank}(A)$. *Then*

$$\min_{X, \mathrm{rank}(X)=r} ||A - X||_2 = \sigma_{r+1} \tag{5.6}$$

holds. A minimum is obtained for $B = U_r\Sigma_r V_r^T = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, *where* U_r, Σ_r, *and* V_r *are partitions of* U, Σ, *and* V *as in (5.4).*

Proof: See proof of Theorem 2.5.3 in [23]. ∎

The above stated theorem is also valid for the Frobenius norm, for which (5.6) is replaced by

$$\min_{X, \mathrm{rank}(X)=r} ||A - X||_F = \left(\sum_{j=r+1}^{\min(m,n)} \sigma_j^2(A) \right)^{\frac{1}{2}}. \tag{5.7}$$

A proof for this can be found in [60].

5.2 Projections

In this section, which is based on [54] and [6], projections are addressed. Before projections are defined, it shall be mentioned that this is done for Hilbert spaces. The reason for doing so in this generality is that proper orthogonal decomposition shall be derived for Hilbert spaces to show that POD works both for the discrete flow vectors as well as for the actual physical quantities, which can be modeled by functions. Note that Hilbert spaces provide a general framework for both settings.

The reader is referred to Section 1.3 of [6] for an introduction to Hilbert spaces. Here, it shall only be mentioned that they are linear spaces together with a scalar product $(\cdot,\cdot)$. The scalar product allows the introduction of the notion of orthogonality, which is important for projections.

A linear operator $\Pi : \mathcal{H} \to \mathcal{H}$ which maps a Hilbert space $\mathcal{H}$ onto itself such that

$$\Pi^2 = \Pi,$$

is called *projection* or *projection operator*.

Colloquially speaking, this implies that if an element $v \in \mathcal{H}$ is projected onto an element $v_1 = \Pi v$, then further projection does not alter the result, since $\Pi v_1 = \Pi^2 v = \Pi v = v_1$. From this simple definition some properties of a projection operator follow, which are stated in the following Theorem.

Theorem 5.3 *Let $\mathcal{H}$ be a Hilbert space. Then $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2$ if and only if there is a projection operator Π such that every $v \in \mathcal{H}$ can be written as*

$$v = v_1 + v_2,$$

where $v_1 = \Pi v \in \text{range}(\Pi)$ and $v_2 = (I - \Pi)v \in \text{range}(I - \Pi)$, and additionally such that $\text{range}(\Pi) = \mathcal{H}_1$ and $\text{range}(I - \Pi) = \mathcal{H}_2$ holds.

Proof: See proof of Proposition 3.6.2 in [6]. ∎

The operator Π is said to project v *onto the subspace $\mathcal{H}_1$ parallel to the subspace $\mathcal{H}_2$.*

5.2.1 Orthogonal projections

An important class of projections are those, for which any $v_1 \in \text{range}(\Pi)$ is orthogonal to any $v_2 \in \text{range}(I - \Pi)$, i.e.

$$(v_1, v_2) = 0 \quad \forall v_1 \in \text{range}(\Pi),\ \forall v_2 \in \text{range}(I - \Pi).$$

A projection with this property is called *orthogonal projection*. Projections that are not orthogonal are *oblique*. The two types of projections are illustrated in Figures 5.1 and 5.2.

While the above definition obviously motivates the name of orthogonal projections, the following Theorem provides a property which is easier to verify.

Theorem 5.4 *Let $\mathcal{H}$ be a Hilbert space. Then the projection $\Pi : \mathcal{H} \to \mathcal{H}$ is orthogonal if and only if Π is self-adjoint, that is, $(\Pi v_1, v_2) = (v_1, \Pi v_2)$ for all $v_1, v_2 \in \mathcal{H}$.*

Proof: See Exercise 3.6.6 in [6].

For the Euclidean space $\mathbb{R}^n$ this means that a projection is orthogonal if the projection matrix is symmetric, i.e. $\Pi = \Pi^T$.

Properties of orthogonal projections

In the following two properties of orthogonal projections are described which will be used frequently in this thesis. The first is the Pythagoras' Theorem.

Theorem 5.5 (Pythagoras' Theorem) *Let $\mathcal{H}$ be a Hilbert space with the induced norm $||.||$ and let Π be an orthogonal projection. Then*

$$||v||^2 = ||\Pi v||^2 + ||(I - \Pi)v||^2$$

holds for all $v \in \mathcal{H}$.

Proof: Since Π is orthogonal, $((I - \Pi)v, \Pi v) = (\Pi v, (I - \Pi)v) = 0$ for all $v \in \mathcal{H}$. Therefore

$$\begin{aligned}
||v||^2 &= ||\Pi v + (I - \Pi)v||^2 = (\Pi v + (I - \Pi)v, \Pi v + (I - \Pi)v) \\
&= (\Pi v, \Pi v) + ((I - \Pi)v, \Pi v) + (\Pi v, (I - \Pi)v) + ((I - \Pi)v, (I - \Pi)v) \\
&= (\Pi v, \Pi v) + ((I - \Pi)v, (I - \Pi)v) = ||\Pi v||^2 + ||(I - \Pi)v||^2. \quad \blacksquare
\end{aligned}$$

Secondly, the following Theorem states that an orthogonal projection projects any $v \in \mathcal{H}$ onto exactly that element of a closed subspace $\mathcal{H}_1$ which has the shortest distance to v.

Theorem 5.6 *Let $\mathcal{H}_1$ be a closed subspace of the Hilbert space $\mathcal{H}$ and $\Pi : \mathcal{H} \to \mathcal{H}_1$ be an orthogonal projection. Then*

$$||v - \Pi v|| = \inf_{v_1 \in \mathcal{H}_1} ||v - v_1||$$

holds for all $v \in \mathcal{H}$.

Proof: See Exercise 3.6.6 in [6].

The statement of Theorem 5.6 is illustrated in Figures 5.1 and 5.2. The orthogonal projection of an element v, given by $\Pi^{\perp}\boldsymbol{v}$, is obviously a better approximation to v than an oblique projection $\Pi\boldsymbol{v}$.

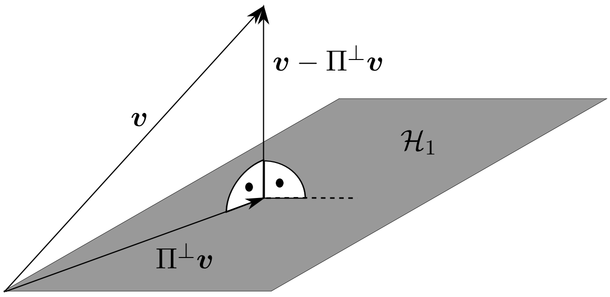


Figure 5.1: *An orthogonal projection.*

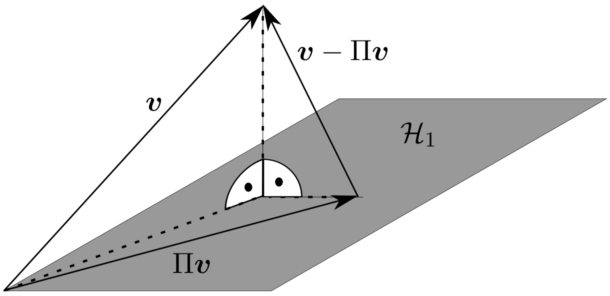


Figure 5.2: *An oblique projection.*

5.2.2 Matrix representations of projections

Consider $\mathcal{H} = \mathbb{R}^n$. Theorem 5.3 implies that for any $\boldsymbol{v} \in \mathbb{R}^n$, the vector $\boldsymbol{u} = \Pi\boldsymbol{v}$ satisfies

$$\begin{aligned} \boldsymbol{u} &\in \operatorname{range}(\Pi), \\ \boldsymbol{v} - \boldsymbol{u} &\in \operatorname{range}(I - \Pi). \end{aligned}$$

If the projection matrix Π has rank d, then the range of $I - \Pi$ is of rank $n - d$. The orthogonal complement of the range of $I - \Pi$, defined by

$$\operatorname{range}(I - \Pi)^{\perp} = \{\boldsymbol{v} \in \mathbb{R}^n : \ (\boldsymbol{v}, \boldsymbol{v}_i) = 0 \ \ \forall \boldsymbol{v}_i \in \operatorname{range}(I - \Pi)\},$$

has dimension d and is used to define range$(I - \Pi)$. With this, the above conditions become (see [54])

$$\begin{aligned} \boldsymbol{u} &\in \operatorname{range}(\Pi), && (5.8) \\ \boldsymbol{v} - \boldsymbol{u} &\perp \operatorname{range}(I - \Pi)^{\perp}. && (5.9) \end{aligned}$$

Equation (5.8) describes the subspace in which the projection resides, while (5.9) gives d restrictions that determine the exact location within the subspace.

Let $U = \begin{bmatrix} \boldsymbol{u}_1 & \dots & \boldsymbol{u}_d \end{bmatrix}$ be a basis for the subspace $\mathcal{H}_1 = \operatorname{range}(\Pi)$ and $W = \begin{bmatrix} \boldsymbol{w}_1 & \dots & \boldsymbol{w}_d \end{bmatrix}$ be a basis for the subspace $\mathcal{H}_2^{\perp} = \operatorname{range}(I - \Pi)^{\perp}$. With the basis U, $\Pi\boldsymbol{v}$ can be exactly represented by $\Pi\boldsymbol{v} = U\boldsymbol{a}$, where $\boldsymbol{a} \in \mathbb{R}^d$ are suitable coefficients. The constraint $\boldsymbol{v} - \Pi\boldsymbol{v} \perp \operatorname{range}(I - \Pi)^{\perp}$ is equivalent to

$$(\boldsymbol{v} - U\boldsymbol{a}, \boldsymbol{w}_j) = 0 \quad \text{for } j = 1, \dots, d$$

which is in matrix form

$$W^T(\boldsymbol{v} - U\boldsymbol{a}) = 0. \tag{5.10}$$

Assuming the matrix (W^TU) is nonsingular, a solution to the above system is given by $\boldsymbol{a} = (W^TU)^{-1}W^T\boldsymbol{v}$. This implies $\Pi\boldsymbol{v} = U\boldsymbol{a} = U(W^TU)^{-1}W^T\boldsymbol{v}$ and the projection is thus represented by the matrix

$$\Pi = U(W^TU)^{-1}W^T. \tag{5.11}$$

In fact, Π is a projection, since

$$\Pi^2 = U(W^TU)^{-1}W^TU(W^TU)^{-1}W^T = U(W^TU)^{-1}W^T = \Pi.$$

In [54] it is stated that if no vector in range(Π) is orthogonal to range$(I - \Pi)^{\perp}$, then (W^TU) is nonsingular.

In the special case that the two bases are *biorthogonal*, that is $(\boldsymbol{u}_i, \boldsymbol{w}_j) = \delta_{ij}$ for $i, j = 1, \ldots, d$ or $W^TU = I$ in matrix form, the projection is given by

$$\Pi = UW^T.$$

Chapter 6

An introduction to projection-based nonlinear model order reduction

In many industrial applications, models of large complexity have to be solved repeatedly. For example, for the simulation of the flow field around an aircraft, differential systems with tens of millions of unknowns have to be solved. Such systems are modeled by the discrete Euler or Navier-Stokes equations, which have been presented in equation (4.3) in Chapter 4. The number of unknowns of these equations are given by the number of variables times the number of grid points. Due to the complex geometry of airplanes, millions of grid points are needed to accurately simulate its aerodynamics.

Obviously, computing the solution of such large systems takes a lot of time. Depending on the computer being used, hours or days will be spent on obtaining a solution. If the problem has to be solved repeatedly for different parameters like the angle of attack or the Mach number, the time issue gets even worse. Sometimes the problem at hand will simply be intractable.

Besides time, also memory is an issue. Models of high order cannot be solved on small machines due to memory shortcomings, but have to be computed on supercomputers. Yet, even supercomputers will sometimes lack enough memory to tackle very complex problems.

Model order reduction (MOR) aims at resolving these issues by replacing the original model with a reduced order model (ROM). As the name suggests, the reduced order model is of smaller complexity which means that fewer unknowns have to be computed. As a result the simulation time and memory requirements will be lowered. Note that the accuracy of the ROM depends on the complexity. Hence, the complexity has to be chosen such that a compromise between accuracy and simulation time is reached.

While a lot of progress has been made in model order reduction for linear problems, many questions remain unanswered for nonlinear problems as pointed out in [1, Section 14.2]. An insight of the difficulties of nonlinear MOR and proposed methods for overcoming these difficulties are given in Section 6.2.

6.1 Basic principles of model order reduction

The goal of model order reduction (MOR) is to replace the original model describing a particular phenomenon by a reduced order model, which consists of far fewer degrees of freedom than the original model. In general this is achieved by two basic principles: On the one hand the solutions of the governing equations are approximated in a *low-dimensional subspace*. On the other hand there is *projection* of the original model. In the following these two concepts are explained. This section is based on [1].

6.1.1 Approximation of the solutions in a low-dimensional subspace

A fundamental idea of model order reduction is to find an appropriate basis for representing the solution of the given model. The basis should have the property that the solutions $\boldsymbol{w} \in \mathbb{R}^N$, which are to be computed, can be approximated well by few ($d \ll N$) basis vectors:
Suppose that a change of basis U is given by

$$U = \begin{pmatrix} U_d & \bar{U} \end{pmatrix} \in \mathbb{R}^{N \times N} \tag{6.1}$$

with $U_d \in \mathbb{R}^{N \times d}$ and $\bar{U} \in \mathbb{R}^{N \times (N-d)}$. Then the desired property is that

$$\boldsymbol{w} = U \cdot \boldsymbol{a}_U = \begin{pmatrix} U_d & \bar{U} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{a} \\ \bar{\boldsymbol{a}} \end{pmatrix} = U_d\, \boldsymbol{a} + \bar{U}\bar{\boldsymbol{a}} \approx U_d\, \boldsymbol{a} \tag{6.2}$$

holds for each solution $\boldsymbol{w} \in \mathbb{R}^N$, where $\boldsymbol{a}_U \in \mathbb{R}^N$, $\boldsymbol{a} \in \mathbb{R}^d$ and $\bar{\boldsymbol{a}} \in \mathbb{R}^{(N-d)}$ are appropriate coefficient vectors. In other words, the change of basis U can be truncated to contain only the most relevant basis vectors, U_d, without losing considerable accuracy for representing the solution.

6.1.2 Approximation by projection

The second basic principle aims at reducing the number of equations by a projection method. Consider the model to be a system of ordinary differential equations (ODE)

$$\frac{\partial \boldsymbol{w}}{\partial t} = \boldsymbol{f}(\boldsymbol{w}), \qquad \boldsymbol{w}(0) = \boldsymbol{w}_0, \tag{6.3}$$

where $\boldsymbol{w}, \boldsymbol{w}_0 \in \mathbb{R}^N$ and $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$.

Let

$$U^{-1} = \begin{pmatrix} W_d^T \\ \bar{W}^T \end{pmatrix} \in \mathbb{R}^{N \times N}$$

with $W_d^T \in \mathbb{R}^{d \times N}$ and $\bar{W}^T \in \mathbb{R}^{(N-d) \times N}$ be the inverse of the change of basis U, which maps the solutions $\boldsymbol{w} \in \mathbb{R}^N$ to the coefficient vectors with respect to the basis given in (6.1). This means that

$$\begin{pmatrix} \boldsymbol{a} \\ \bar{\boldsymbol{a}} \end{pmatrix} = \begin{pmatrix} W_d^T \\ \bar{W}^T \end{pmatrix} \cdot \boldsymbol{w}.$$

Since $U^{-1}U = I$ and thus $W_d^T U_d = I_d$, it readily follows that $\Pi = U_d W_d^T \in \mathbb{R}^{N \times N}$ is a projection.

By applying $\Pi = U_d W_d^T$ to the original model (6.3), the ODE system is projected onto the subspace spanned by U_d along the kernel of W_d^T. This leads to

$$\Pi \cdot U \cdot \left(\frac{\partial \boldsymbol{a}_U}{\partial t} \right) = \Pi(\boldsymbol{f}(U\boldsymbol{a}_U)) \tag{6.4}$$

$$\Longleftrightarrow \; U_d \cdot \frac{\partial \boldsymbol{a}}{\partial t} = U_d W_d^T \cdot \boldsymbol{f}\left(U_d \boldsymbol{a} + \bar{U}\bar{\boldsymbol{a}}\right). \tag{6.5}$$

Note that if $W_d = U_d$, the projection is called *Galerkin projection*. Otherwise the projection is referred to as *Petrov-Galerkin projection*. If (6.2) is assumed and the term $\bar{U}\bar{\boldsymbol{a}}$ is neglected, then equation (6.5) multiplied with W_d^T becomes

$$\frac{\partial \boldsymbol{a}}{\partial t} = W_d^T \cdot \boldsymbol{f}\left(U_d \boldsymbol{a}\right). \tag{6.6}$$

Note that the ODE system (6.6) can be interpreted as the transformation of equation (6.3) into the coefficient space of the basis U_d. It consists of as many unknowns as equations and can thus be solved on its own. Its complexity of d is typically much smaller than the complexity of N of the original model (6.3). Thus, it is called *reduced system.*

With the solution of the reduced system $\boldsymbol{a}$, a solution to the original model (6.3) is obtained with $\boldsymbol{w} \approx U_d \boldsymbol{a}$.

6.2 The problem of nonlinear model order reduction

In case of a linear model, the function $\boldsymbol{f}$ in equation (6.6) is linear and can be seen as a matrix $A \in \mathbb{R}^{N \times N}$. Then (6.6) becomes

$$\frac{\partial \boldsymbol{a}}{\partial t} = W_d^T A U_d \boldsymbol{a}, \tag{6.7}$$

where $\tilde{A} := W_d^T A U_d \in \mathbb{R}^{d \times d}$ is a matrix of the size of the reduced number of equations d and can be computed once when setting up the reduced model. Thus the obtained reduced system is computationally inexpensive to solve.

The problem in case of nonlinear model order reduction is that although the reduced model (6.6) contains only few equations, the right hand side is computationally expensive to evaluate. This is due to the fact that the function $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ of the order N of the original model has to be computed.

In order to overcome this problem, various methods have been proposed. Rewienski et al. [48, 49, 62] suggest to linearize the function $\boldsymbol{f}$ with Taylor series expansion at various linearization points. The various linearized models are then combined by a weighted sum of them to obtain a reduced order model. This so-called Trajectory-Piecewise Linear (TPWL) approach is especially promising for weakly nonlinear models. In [24] the approach is applied to an actively controlled supersonic diffuser using the Euler equations.

Besides linearization of the right hand side as in TPWL, other approaches aim at deriving models which do not require the evaluation of all N entries of the vector-valued function $\boldsymbol{f}$.

Rather, only $n \ll N$ entries of $\boldsymbol{f}$ need to be computed for solving the reduced model. Among such kind of approaches are the missing point estimation [5] and the discrete empirical interpolation method [13].

Chapter 7

Proper orthogonal decomposition

In the previous chapter it has been outlined that the two basic principles of model order reduction are the approximation in a suitable low-dimensional subspace and the reduction of the number of equations by projection. Proper orthogonal decomposition, abbreviated by POD, is concerned with the first principle. That is, it is a technique to find a low-dimensional subspace which is capable of accurately representing solutions of the governing equations.

Given a set of snapshots, which are solutions of the governing equations characteristic of the model to be constructed, POD extracts the most dominant features, called modes. These modes form a basis for a low-dimensional subspace of the predominant dynamics of the system. That is, discrete solutions $\boldsymbol{w} \in \mathbb{R}^N$ of a system of the form (6.3) can be represented by

$$\boldsymbol{w} = U_d\, \boldsymbol{a}(t) + \bar{\boldsymbol{w}}.$$

In contrast to the representation (6.1) a shift $\bar{\boldsymbol{w}}$ has been introduced. The shift is described in detail in Section 7.3. For the time being, suppose that there is no shift, i.e. $\bar{\boldsymbol{w}} = \mathbf{0}$.

If the predominant dynamics of the system is captured in the modes, it is assumed that solutions to the governing equations – even those not included in the snapshots – can be constructed with the modes U_d, also called POD basis. Obviously, in order for this assumption to be valid, the snapshots have to be chosen carefully such that all the relevant dynamics can be extracted.

Although proper orthogonal decomposition is often used to obtain reduced order models for non-linear problems, POD itself is a linear method. That is, the low-dimensional subspace obtained by POD is linear. In [26, p. 87] it is emphasized that

> “[l]inearity is the source of the method’s strength as well as its limitations.”

What is meant by this, is that on the one hand, linearity allows the use of a vast set of tools from mathematics such as the theory of linear operators. With these tools it can be shown that POD is optimal in the sense that no other linear subspace is able to capture more energy on average than the POD basis [26, p. 98]. Yet, at the same time, one has to keep in mind that this optimality is limited to linear subspaces and that non-linear approaches might perform significantly better.

In [31, 26] and the references therein, it is documented that POD has been successfully applied to many different fields of research such as fluid mechanics, structural mechanics, random variables, image processing, meteorology, and oceanography, to name but a few examples. Note that proper orthogonal decomposition is also called Karhunen-Loève decomposition, principal component analysis, and singular value decomposition by different scientific communities.

The close connection to the singular value decomposition will be addressed in the first section. In this section the easiest, but at the same time the numerically most relevant case of discretized solutions of partial differential equations living in finite-dimensional vector spaces is tackled to show the simplicity of the POD technique.

In the subsequent section a more general derivation of POD is given, in which general Hilbert spaces are considered. This derivation requires knowledge of the properties of linear operators in Hilbert spaces. The generality of this framework has the advantage that on the one hand it covers the discrete L_2 Hilbert space used later on in this thesis (see Section 9.2). On the other hand, it also shows that POD works for continuous spaces like the L_2 function space.

7.1 POD for the discrete Euclidean space

At first, the proper orthogonal decomposition will be outlined for a finite dimensional vector space with an Euclidean scalar product. This is the easiest setting and shall thus be at the center of attention first. Consider the vector space to be the one, which is obtained from spatial discretization of a partial differential equation (PDE). Thus each component of a vector represents the value of the solution of the PDE at a specific grid point.

Assume there is a set of *snapshots* $\boldsymbol{w}_i \in \mathbb{R}^N$ with $i = 1, \ldots, m$, which are solutions to the PDE for different time instances or for different parameter values. Later in this thesis the snapshots are steady state flow solutions of the discretized Euler and Navier-Stokes equations given by (4.11) for varying parameter values of the angle of attack and the freestream Mach number. Note, however, that flow solutions consist of more than one variable, which makes the derivation of POD more technical. For this reason, the derivation of POD for flow problems shall be postponed until Chapter 9. For now, assume that the snapshots are of the simple form $\boldsymbol{w}_i \in \mathbb{R}^N$ and that they are solutions to the system (6.3). Consider them to be stored in a matrix $Y = \left[\begin{array}{ccc} \boldsymbol{w}_1 & \cdots & \boldsymbol{w}_m \end{array} \right]$.

The goal of POD is to find a d-dimensional subspace in which the given data is approximated optimally. In other words, a projection Π_d into the subspace is sought, for which

$$\min ||Y - \Pi_d Y||_2 \tag{7.1}$$

holds.

Note that projections onto a d-dimensional subspace can obviously be represented by matrices of rank d. According to Theorem 5.2, a minimum to (7.1) for all such matrices is obtained for the truncated SVD of Y. Consequently, solving (7.1) is equivalent to finding a projection Π_d, which projects Y onto its truncated SVD.

Recall from Section 5.1 that the SVD of Y is given by $Y = U\Sigma V^T$, where U as well as V are orthogonal matrices and Σ is a matrix, whose non-diagonal entries are zero and the diagonal entries are the singular values. Furthermore, recall that the truncated SVD of Y is given by $\tilde{Y} = U_d \Sigma_d V_d^T$, where d is the rank of the matrix $\tilde{Y}$ and U_d as well as V_d are matrices, which consist of only the first d columns of U and V, respectively. Note that due to the construction of U_d the columns of this matrix are orthogonal to those of U, i.e. $U_d^T U = (I_d \;\; \mathbf{0})^T$.

Now focus again on problem (7.1). A solution is given by the projection $\Pi_d = U_d U_d^T$, since Π_d projects Y onto its truncated SVD, as

$$\Pi_d Y = U_d U_d^T U \Sigma V^T = U_d \Sigma V^T = U_d \Sigma_d V_d^T$$

shows. Thus Π_d maps the snapshot matrix Y onto the span of the columns of U_d. Exactly these column vectors – the first d left singular vectors $\boldsymbol{u}_i$ of Y – form the *(truncated) POD basis.*

In conclusion, it can be said that POD basis construction is strongly related to the SVD analysis of the snapshot matrix. Due to this fact it is obvious that the snapshot matrix has to be chosen carefully such that it contains all important features that are to be represented by the basis. Solutions in the POD subspace are given by

$$\boldsymbol{w} = U_d \, \boldsymbol{a}(t).$$

7.2 Derivation of the POD for Hilbert spaces

In this section, POD will be derived for Hilbert spaces $\mathcal{H}$. Special focus will be given to the finite dimensional space $\mathbb{R}^N$ and the functional space L_2, for the latter allows POD analysis for functions rather than discrete approximations, e.g., the conservative variables $\boldsymbol{W}(t)$ solving the *non-discrete* Euler and Navier-Stokes equations (3.23). It will be shown that a necessary condition for the existence of the POD basis is that it solves an eigenvalue problem that is closely related to the SVD in the discrete case.

7.2.1 A necessary condition for the existence of the POD basis

Assume there are snapshots $\{w_k \in \mathcal{H}\}$ available. The linear space spanned by the snapshots shall be denoted by $\mathcal{W}$. Note that it is not specified how many snapshots are given. In particular also infinitely many are allowed. Although this is irrelevant for practical applications, it will yield theoretical insights of POD.

The goal of POD is to find a subspace $\mathcal{U}_d = \{u_j \in \mathcal{H}; j = 1, \ldots, d\}$, such that the orthogonal projection $\Pi_d : \mathcal{W} \to \mathcal{U}_d$ of the snapshots into the sought subspace is optimal in the sense that

$$\inf_{\Pi_d} \operatorname{Avg}\left(||w_k - \Pi_d w_k||^2\right), \tag{7.2}$$

where Avg(.) is an linear averaging operation, which is assumed to commute with the scalar product. In the finite dimensional case this is $1/m \sum_{i=1}^{m}[.]$ for example.

The search for the infimum of the error (7.2) is equivalent to finding the supremum of the projection $\mathrm{Avg}\left(||\Pi_d w_k||^2\right)$. This is due to Pythagoras' theorem for orthogonal projections Π, which states that

$$||w_k||^2 = ||w_k - \Pi w_k||^2 + ||\Pi w_k||^2.$$

At first a single basis element[1] u is sought. Note that in this case the projection is given by $\Pi_1 w_k = (w_k, u)u$. It follows that $\mathrm{Avg}\left(||\Pi_1 w_k||^2\right) = \mathrm{Avg}\left(||(w_k, u)u||^2\right) = \mathrm{Avg}\left(|(w_k, u)|^2 ||u||^2\right)$. The problem (7.2) thus becomes

$$\sup_{u\in\mathcal{H}} \mathrm{Avg}\left(|(w_k, u)|^2\right) \qquad \text{s.t.} \qquad ||u||^2 = 1. \tag{7.3}$$

With the method of the Lagrangian multipliers, the problem can be solved by defining the functional $J[u] = \mathrm{Avg}(|(w_k, u)|^2) - \lambda\left(||u||^2 - 1\right)$. A necessary condition for the solution to (7.3) is then obtained by demanding that the first variation vanishes, i.e.

$$\frac{d}{d\delta} J[u + \delta v]\Big|_{\delta=0} = 0 \qquad \forall u + \delta v \in \mathcal{H}.$$

This yields

$$\begin{aligned}
&\frac{d}{d\delta} J[u + \delta v]\Big|_{\delta=0} \\
&= \frac{d}{d\delta}\Big[\mathrm{Avg}\left[(w_k, u + \delta v)(u + \delta v, w_k)\right] - \lambda\{(u + \delta v, u + \delta v) - 1\}\Big]\Big|_{\delta=0} \\
&= \frac{d}{d\delta}\Big[\mathrm{Avg}[(w_k, u)(u, w_k)] + \delta\mathrm{Avg}[(w_k, v)(u, w_k)] + \delta\mathrm{Avg}[(w_k, u)(v, w_k)] \\
&\qquad + \delta^2\mathrm{Avg}[(w_k, v)(v, w_k)] - \lambda\{(u, u) + \delta(u, v) + \delta(v, u) + \delta^2(v, v) - 1\}\Big]\Big|_{\delta=0} \\
&= \mathrm{Avg}\left[(w_k, v)(u, w_k) + ((w_k, v)(u, w_k))^*\right] - \lambda\left[(u, v) + (u, v)^*\right] \\
&= 2\mathrm{Re}\{\mathrm{Avg}[(w_k, v)(u, w_k)] - \lambda(u, v)\} = 0.
\end{aligned}$$

With a calculation for $u + \delta \mathrm{i} v \in \mathcal{H}$ in an analogous manner

$$\begin{aligned}
\frac{d}{d\delta} J[u + \delta \mathrm{i} v]\Big|_{\delta=0} &= \mathrm{Avg}\left[-\mathrm{i}(w_k, v)(u, w_k) + \mathrm{i}\left((w_k, v)(u, w_k)\right)^*\right] - \lambda\left[-\mathrm{i}(u, v) + \mathrm{i}(u, v)^*\right] \\
&= 2\mathrm{Im}\{\mathrm{Avg}[(w_k, v)(u, w_k)] - \lambda(u, v)\} = 0,
\end{aligned}$$

is obtained. Thus $\mathrm{Avg}[(w_k, v)(u, w_k)] - \lambda(u, v) = 0$.

Since

$$\begin{aligned}
0 &= \mathrm{Avg}[(w_k(u, w_k), v)] - \lambda(u, v) = (\mathrm{Avg}[w_k(u, w_k)], v) - \lambda(u, v) \\
&= (\mathrm{Avg}[w_k(u, w_k)] - \lambda u, v)
\end{aligned}$$

for all $v \in \mathcal{H}$, the eigenvalue problem

$$Ru = \lambda u \tag{7.4}$$

is obtained, where R is the operator defined by $\mathrm{Avg}[w_k(u, w_k)]$.

To sum up this subsection, it has been shown that a solution u to (7.3) necessarily has to be an eigenelement of the eigenvalue problem (7.4). In other words, due to the equivalence of problems (7.3) and (7.2), a necessary condition for the existence of a POD basis element u is that it solves the eigenvalue problem (7.4).

---

[1] In $\mathbb{R}^N$ the element is a vector and in L_2 this is a function.

7.2.2 A sufficient condition for the existence of the POD basis

For a general Hilbert space, a sufficient condition can be derived under certain assumptions by investigating the operator $Ru = \mathrm{Avg}[w_k(u, w_k)]$. This is due to fact that the original problem (7.3) can be reformulated with the help of R as the following computation shows:

$$(Ru, u) = (\mathrm{Avg}[w_k(u, w_k)], u) = \mathrm{Avg}[(u, w_k)(w_k, u)] = \mathrm{Avg}\left(|(w_k, u)|^2\right). \tag{7.5}$$

Thus the search of the supremum in (7.3) is equivalent to finding the supremum of (Ru, u), which is a well studied problem in operator theory.

Before tackling this problem, note that R is linear, self-adjoint, and under certain conditions compact. The linearity follows directly from the definition of the scalar product. Furthermore, it is easily verified that R is self-adjoint, as the following calculation for all $x, z \in H$ shows:

$$\begin{aligned} (Rx, z) &= (\mathrm{Avg}[w_k(x, w_k)], z) = \mathrm{Avg}[(x, w_k)(w_k, z)] \\ &= \mathrm{Avg}[(x, w_k)(z, w_k)^*] = (x, \mathrm{Avg}[w_k(z, w_k)]) = (x, Rz). \end{aligned}$$

The proof of compactness of R is more difficult. The reader is referred to [26] to find a proof for the L_2 space under certain conditions.

In Chapter 6 of [50] it is proven that for linear, self-adjoint, and compact operators a maximum equal to the greatest eigenvalue λ exists and is obtained for the corresponding eigenfunction u_1 of R. Due to equation (7.5) u_1 is also the solution to (7.3).

In a next step, problem (7.3) is solved with the additional restriction that the solution u is now orthogonal to u_1. It is shown in [50] that the maximum is equal to the second largest eigenvalue of R and is obtained for the corresponding eigenfunction u_2. This procedure can be repeated until a d-dimensional subspace of the first eigenfunction u_j is found.

To sum up, it has been shown that in an average sense the best approximation of all $w_k \in \mathcal{W}$ by an orthogonal projection onto a d-dimensional subspace is given by

$$\Pi_d w_k = \sum_{j=0}^{d} (w_k, u_j) u_j = \sum_{j=0}^{d} a_j u_j,$$

where a_j are the so-called *modal coefficients* or *POD coefficients* and u_j are the eigenfunctions of R, which are referred to as the *POD basis*.

Note that if R is compact for any Hilbert space $\mathcal{H}$, then this statement is also valid for $\mathcal{H}$. This is because, in the above, only the proof of compactness is specific to the L_2. In particular, note that if R is represented by a finite-dimensional matrix, it is a compact operator.

The discrete Euclidean space

Although it has already been shown in Section 7.1 that for the finite dimensional case, the POD basis is given by the left singular vectors, it will be outlined here that the eigenvalue problem (7.4) yields the same results.

Suppose that m snapshots $\boldsymbol{w}_k$ are given and that the average will be determined by $\sum_{i=1}^{m} 1/m\,[.]$. For the finite dimensional space $\mathbb{R}^n$ the eigenvalue problem (7.4) becomes

$$\begin{aligned}\lambda \boldsymbol{u} &= R\boldsymbol{u} = \operatorname{Avg}[\boldsymbol{w}_k(\boldsymbol{u},\boldsymbol{w}_k)] = \sum_{k=1}^{m} \frac{1}{m}(\boldsymbol{u}^T\boldsymbol{w}_k)\cdot \boldsymbol{w}_k \\ &= \frac{1}{m}\begin{bmatrix} \boldsymbol{w}_1 & \cdots & \boldsymbol{w}_m \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{w}_1^T\boldsymbol{u} \\ \vdots \\ \boldsymbol{w}_m^T\boldsymbol{u} \end{bmatrix} \\ &= \frac{1}{m} YY^T \cdot \boldsymbol{u},\end{aligned}$$

In Subsection 5.1.1 it was outlined that the eigenvectors of YY^T are the left singular vectors of Y. Obviously, a solution to (7.3) is given by the left singular vector $\boldsymbol{u}_1$ corresponding to the greatest singular value. This follows from Theorem 5.2.

Next equation (7.3) is solved with the further restriction that the solution u is orthogonal to $\boldsymbol{u}_1$. Obviously, $\boldsymbol{u}$ is the left singular vector corresponding to the second largest singular value. This procedure is iterated until the first d vectors are found.

Clearly, the derivation of POD is less complicated directly via the SVD and its properties as shown in Section 7.1.

7.3 POD for affine subspaces

While the derivation of POD is done for linear subspaces in the previous sections, it shall be highlighted here that the derivations are also valid for affine subspaces. In fact, considering affine subspaces introduces only a shift to the snapshots before the POD is computed and afterwards a backshift of the solution:

Let the shift be denoted by $\bar{\boldsymbol{w}}$. Then in the Euclidean space the snapshot matrix is redefined as

$$Y = [(\boldsymbol{w}_1 - \bar{\boldsymbol{w}}) \ \ldots \ (\boldsymbol{w}_M - \bar{\boldsymbol{w}})] \in \mathbb{R}^{N\times M}. \tag{7.6}$$

After computing the POD, one obtains a basis U_d, which is only capable of representing shifted solutions. Hence, in order to get solutions similar to the snapshots, the backshift has to be applied. That is,

$$\boldsymbol{w} = U_d\, \boldsymbol{a}(t) + \bar{\boldsymbol{w}}, \tag{7.7}$$

where $\boldsymbol{a}(t)$ is a vector of suitable POD coefficients.

Shifting by the average: A commonly used shift is the average of the snapshot ensemble [47, 31, 45], which is given by

$$\bar{\boldsymbol{w}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{w}_i. \tag{7.8}$$

In [72, Subsection 2.3] it is suggested that this shift is used in context of CFD.

First of all, this is due to the fact that some of the physical quantities in CFD such as density and pressure are necessarily strictly positive. This implies that these quantities do not lie in a vector space. Note, however, that a vector space is a prerequisite for a Hilbert space and thus, along with the assumption that the snapshots lie in a Hilbert space, also the derivation in the Section 7.2 is compromised. The fluctuations around a mean, however, may feature both positive and negative values, which motivates the use of an affine subspace.

Secondly, it has been discovered in practical experiments [72] that shifting the solutions with the average reduces the occurrence of physically infeasible solutions.

Another advantage of an affine subspace with this kind of shift is that instead of having to model the dynamics as a whole, only the deviation from the mean has to be accounted for. This is visualized in Figure 7.1, where the red dot is the average and the blue ones are the snapshots. Obviously, if the snapshots are centered around the origin as in Figure 7.1(a), a linear subspace is well suited. However, if the snapshots are far from the origin (see Figure 7.1(b)), then an affine subspace is better.

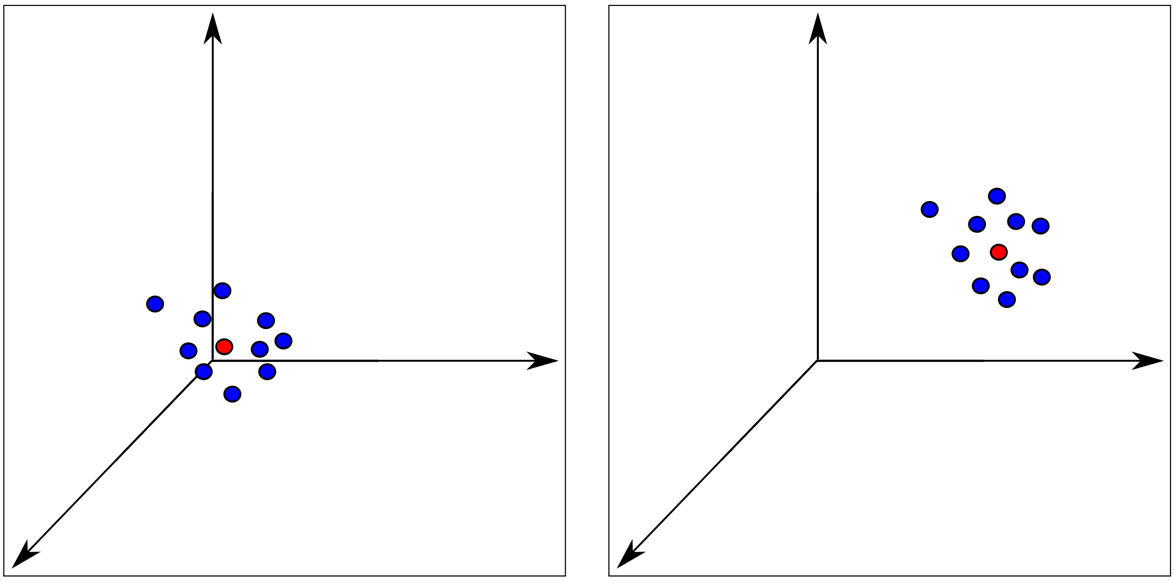

(a) A snapshot set suited for a linear subspace. (b) A snapshot set suited only for an affine subspace.

Figure 7.1: *Two different snapshot sets, where the red dot is the average and the blue ones are the snapshots.*

The following Lemma proves that the snapshot matrix does not have full rank, if it is shifted. This can simply be seen by the fact that the sum of the columns is zero and thus the columns are linearly dependent. In the proof of the Lemma, this argument is presented more formally.

Lemma 7.1 *Let* $\bar{\boldsymbol{w}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{w}_i$ *and* $Y = [\,(\boldsymbol{w}_1 - \bar{\boldsymbol{w}}) \;\cdots\; (\boldsymbol{w}_m - \bar{\boldsymbol{w}})\,] \in \mathbb{R}^{N\times m}$ *with* $N \geq m$*. Then the rank of* Y *is smaller than* m*.*

Proof: In order to show that the rank of Y is smaller than m, it suffices to prove that the columns of Y are linearly dependent, that is, $\gamma_1, \ldots, \gamma_m \in \mathbb{R}$ exist, where at least one $\gamma_i \neq 0$, such that

$$\gamma_1(\boldsymbol{w}_1 - \bar{\boldsymbol{w}}) + \ldots + \gamma_m(\boldsymbol{w}_m - \bar{\boldsymbol{w}}) = \boldsymbol{0}.$$

This is equivalent to

$$\sum_{i=1}^{m} \left(\gamma_i - \frac{1}{m} \sum_{j=1}^{m} \gamma_j \right) \boldsymbol{w}_i = \boldsymbol{0}. \tag{7.9}$$

Note that for $\gamma_i = 1$ for all $i \in \{1, \ldots, m\}$, it holds that

$$\gamma_i = \frac{1}{m} \sum_{j=1}^{m} \gamma_j.$$

Hence, equation (7.9) is satisfied and $\gamma_i \neq 0$ not just for one i, but in fact for all $i = 1, \ldots, m$. Therefore, the columns of Y are linearly dependent and the rank of Y is at least smaller than m. ∎

Note that the above Lemma has important implications for the use of the POD basis: Since the shifted snapshot matrix does not have full rank, the smallest singular value is exactly zero, i.e. $\sigma_m = 0$. Therefore, the corresponding singular vector $\boldsymbol{u}_m$ does not carry any information whatsoever about the snapshots and it should always be disregarded.

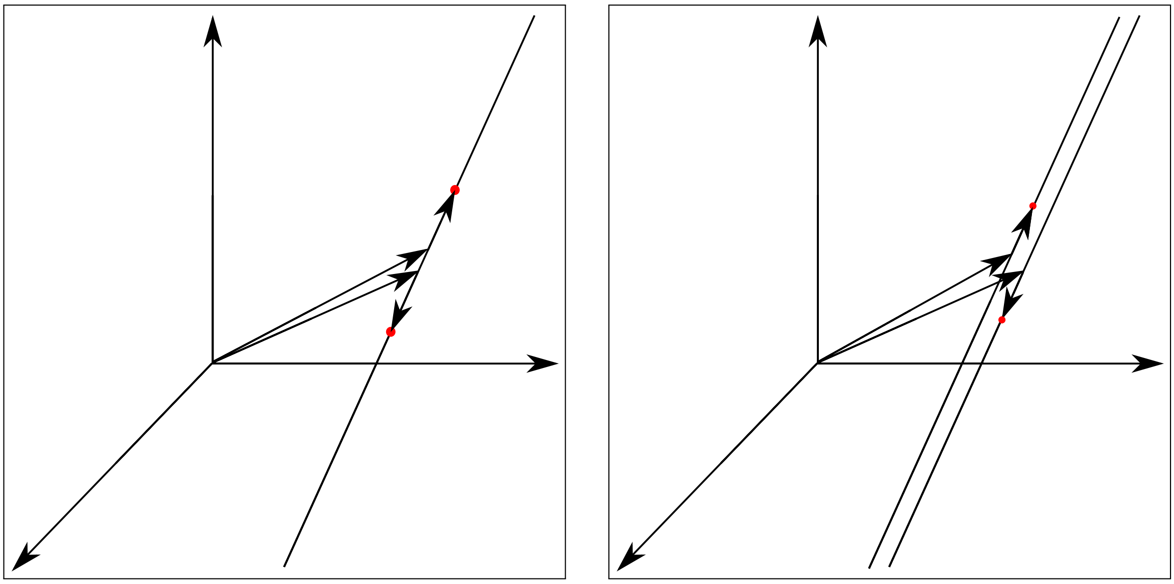
As a result of this, with an affine subspace only $m - 1$ basis vectors are needed to exactly reconstruct any snapshot included in the snapshot matrix. A linear subspace, on the other hand, would require m snapshots. This suggests that the average is a mode itself and that, in general, less basis vectors might be needed for affine POD representations.

Shifting by freestream values: Another possible shift is given by the freestream values for the considered flow as suggested in [40]. In this case the deviation from the freestream condition is modeled. As the name suggests, the freestream condition is the flow at the same velocity and angle, which is not disturbed by any object.

Since freestream values are prescribed at the farfield boundary for supersonic inflow (see Section 4.4), subtracting the freestream values from the snapshots in (7.6) yields zero-valued farfield inflow boundaries in the POD basis [40]. This has the obvious advantage that after adding the values again in (7.7), the farfield boundary conditions are matched exactly for supersonic flows at the inflow.

Note that the freestream values are dependent on the angle of attack and the Mach number of the flow (see Section 4.4). Thus, with this shift, matching supersonic inflow boundary conditions exactly can be guaranteed, even for changing freestream conditions. However, note that in this case, the shift $\bar{\boldsymbol{w}}$ is different for each snapshot and each sought solution, if the flow condition is different. This can result in different POD representations of the form (7.7) to not lie in the same affine subspace. In Figure 7.2(b) this scenario is sketched.

If, however, the Mach number and the angle of attack change only slightly, then also the freestream values are fairly similar. Hence, it can be assumed that with a moderate change of



(a) The scenario when different freestream conditions lie in the same affine subspace.

(b) The scenario when different freestream conditions do not lie in the same affine subspace.

Figure 7.2: *The two possible scenarios when shifting with different freestream conditions. The lines represent the linear subspace spanned by the POD basis, i.e.* $\operatorname{span}(U_d)$*, while the vectors to the lines sketch different shifts* $\bar{\boldsymbol{w}}_1$ *and* $\bar{\boldsymbol{w}}_2$*. The vectors on the lines stand for* $U_d\boldsymbol{a}_1$ *and* $U_d\boldsymbol{a}_2$ *and the red dots for the actual solutions.*

the flow conditions, the POD representations of the form (7.7) lie approximately in the same affine subspace as shown in Figure 7.2(a).

Since supersonic flows will not particularly be of interest in this thesis, the just described shifting procedure is not considered for the implementation of POD.

7.4 Summary

In this final section of the chapter, a brief overview of POD shall be given. The starting point of POD is to assume that snapshots of the governing equations are given. It shall be highlighted once again that the snapshots have to be representative of the solutions, which shall be obtained from the reduced order model. That is, if some features of the sought after solutions are not obtained in the snapshots, then the POD representation of the solutions are not able to accurately approximate solutions with such features.

Algorithm 1 summarizes the POD methodology. The snapshots are the input to it. They are stored in a matrix. In case an affine POD is used, the shift is subtracted from each snapshot vector. Subsequently, the singular value decomposition (SVD) of the snapshot matrix is computed. This yields the left singular vectors u_i, stored in U. From these vectors u_i only the d most relevant column vectors are selected. For example, only those singular vectors are chosen, which correspond to the largest singular values. The selection of the singular vectors finally yields the truncated POD basis $U_d = (u_1, \dots, u_d)$. With the help of this basis, solutions in the POD subspace are then represented by $\boldsymbol{w} = U_d\, \boldsymbol{a}(t)$ or $\boldsymbol{w} = U_d\, \boldsymbol{a}(t) + \bar{\boldsymbol{w}}$, depending on the fact if an affine shift is considered or not.

Algorithm 1 Proper orthogonal decomposition

```
Input: Snapshots $y_1, \dots, y_m$
Output: POD basis vectors $u_1, \dots, u_d$
Store all snapshots in the snapshot matrix $Y = (y_1, \dots, y_m)$.
Compute the singular value decomposition (SVD) of the snapshot matrix $Y = U\Sigma V$.
Truncate the POD basis such that the $d$ most relevant are kept, that is, $U_d = (u_1, \dots, u_d)$.
```

Chapter 8

Missing point estimation

As it has been stated in Section 6.2, missing point estimation (MPE) is a model order reduction technique, which is well suited for nonlinear problems. While POD is a method for constructing a suitable basis, MPE is an approach for obtaining a reduced order model based on projection as described in Subsection 6.1.2.

Assume the original model to be of the form (6.3). In the later chapters this will eventually be the discretized Euler or Navier-Stokes equations (4.3). The projection of MPE avoids the costly evaluation of the right hand side $\boldsymbol{f}^T = (f_1, \dots, f_N)$ at *all* components, that is, at all f_i with $i = 1, \dots, N$. Instead, only some components f_i are carefully selected, at which the right hand side is evaluated. Obviously, this leads to higher numerical efficiency.

Missing point estimation is based on the idea that, in principle, only d pieces of information contained in the right hand side suffice to find the d POD coefficients corresponding to the basis vectors. Clearly, these pieces of information have to be distinct. To be on the safe side in practice, more than d points will be chosen.

Missing point estimation has been introduced in the PhD thesis of Patricia Astrid [3] and has been documented very well in [5]. Amongst the various applications, that MPE has been applied to, are heat transfer processes [5, 2], electrical circuit modeling [4, 68], and oil reservoir simulation [12]. Furthermore the author of this work has applied MPE to aerodynamics [63, 64] and is not aware of any previous application of MPE in this field of research.

8.1 Derivation of the missing point estimation

Consider the system of the original model to be of the form

$$\frac{\partial \boldsymbol{w}}{\partial t} = \boldsymbol{f}(\boldsymbol{w}), \qquad \boldsymbol{w}(0) = \boldsymbol{w}_0. \tag{6.3}$$

As a first step, represent the solution with the help of the reduced POD basis and a shift as in (7.7). If no shift is considered, it can be set to $\bar{\boldsymbol{w}} = \boldsymbol{0}$ throughout. Inserting the representation into the system yields

$$U_d \frac{\partial \boldsymbol{a}}{\partial t} = \boldsymbol{f}\left(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}\right) + \boldsymbol{\epsilon}_0, \tag{8.1}$$

where $\boldsymbol{\epsilon}_0$ is the error which is introduced by this approximation. Note that it is assumed that the shift is time-independent and consequently, $\frac{\mathrm{d}\bar{\boldsymbol{w}}}{\mathrm{d}t} = \mathbf{0}$. This assumption is always valid if the shift is the average of the snapshots, since this average is independent of time.

The idea of missing point estimation is to construct a reduced order model, for which the right hand side $\boldsymbol{f}$ is to be evaluated only at a subset of its components f_i corresponding to particular grid points. This is done due to the fact that the number N is typically very large, at which $\boldsymbol{f} \in \mathbb{R}^N$ has to be evaluated. Hence, reducing the evaluation points will increase the computational efficiency.

Let the subset of selected evaluation points be given by $\mathbb{X} = \{j_1, \ldots, j_n\} \subset \{1, \ldots, N\}$, where n is the number of the selected points. With the definition of the *selection matrix*

$$P = \begin{pmatrix} \boldsymbol{e}_{j_1} & \cdots & \boldsymbol{e}_{j_n} \end{pmatrix} \in \mathbb{R}^{N \times n},$$

where $\boldsymbol{e}_j \in \mathbb{R}^N$ is the jth unit vector, a projection[1] onto the chosen points $\mathbb{X} = \{j_1, \ldots, j_n\}$ can be constructed by $\Pi_P = PP^T$. Applying Π_P to equation (8.1) yields

$$PP^T U_d \frac{\partial \boldsymbol{a}}{\partial t} = PP^T \boldsymbol{f}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}) + PP^T \boldsymbol{\epsilon}_0.$$

Note that at this point there are d unknowns $\boldsymbol{a}$, but n relevant equations. By imposing orthogonality conditions upon the system, the number of equations is reduced to be equal to the number of unknowns. The orthogonality conditions state that the error $PP^T U_d \frac{\partial \boldsymbol{a}}{\partial t} - PP^T f(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})$ is orthogonal to each POD basis vector[2], that is

$$\left(\boldsymbol{u}_j \, , PP^T U_d \frac{\partial \boldsymbol{a}_j}{\partial t} - PP^T \boldsymbol{f}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}) \right) = 0, \qquad j = 1, \ldots, d. \tag{8.2}$$

In matrix notation and for the Euclidean scalar product this reads

$$U_d^T \left(PP^T U_d \frac{\partial \boldsymbol{a}}{\partial t} - PP^T \boldsymbol{f}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}) \right) = 0. \tag{8.3}$$

Note that this is equivalent to $U_d^T PP^T \boldsymbol{\epsilon}_0 = \mathbf{0}$, but this does not mean that the error $\boldsymbol{\epsilon}_0$ is not in the span of the POD basis. Rather, only the projection of the error $PP^T \boldsymbol{\epsilon}_0 \notin \operatorname{span}(U_d)$. For this reason the points of $\mathbb{X} = \{j_1, \ldots, j_n\}$ have to be chosen with care.

The reduced system of the missing point estimation is finally obtained by premultiplying equation (8.3) with $(U_d^T PP^T U_d)^{-1}$ and it reads

$$\frac{\partial \boldsymbol{a}}{\partial t} = (U_d^T PP^T U_d)^{-1} U_d^T PP^T \boldsymbol{f}(U_d \boldsymbol{a}). \tag{8.4}$$

Note that it is of the order $d \ll N$ and thus it is of small complexity. Furthermore, due to $P^T \boldsymbol{f}(U_d \boldsymbol{a})$ only some entries of the right hand side have to be evaluated which will accelerate the computation.

---

[1] The matrix of the projection $\Pi_P = PP^T$ is a diagonal matrix with the diagonal entries $\Pi_{P,ii} = 1$ for $i \in \mathbb{X} = \{j_1, \ldots, j_n\}$. Obviously, $\Pi_P^2 = PP^T PP^T = PP^T = \Pi_P$.

[2] The orthogonality conditions are motivated as follows: The goal is to find a solution of the system (6.3) in the POD subspace, which approximates the actual solution well. A good approximation is given if the error of the system, here $PP^T U_d \frac{\partial \boldsymbol{a}}{\partial t} - PP^T f(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})$, has minimal distance to the POD subspace, i.e. the error of the system with respect to the subspace, in which the solution is sought, is minimal. This is obviously the case if the error is orthogonal to the subspace.

8.2 The projection of the missing point estimation

The derivation of the missing point estimation can also be recasted into the projection framework of Subsection 6.1.2. Consider the projection $\Pi = U_d W_d^T$ in equation (6.4). By comparing equations (6.6) and (8.4), it can easily be seen that $W_d^T = (U_d^T P P^T U_d)^{-1} U_d^T P P^T$ and thus the projection is

$$\Pi = U_d (U_d^T P P^T U_d)^{-1} U_d^T P P^T. \tag{8.5}$$

Note that it is in fact a projection, since

$$\Pi^2 = U_d (U_d^T P P^T U_d)^{-1} U_d^T P P^T U_d (U_d^T P P^T U_d)^{-1} U_d^T P P^T = U_d (U_d^T P P^T U_d)^{-1} U_d^T P P^T = \Pi.$$

In general, Π will not be orthogonal, but oblique, and since $U_d \neq W_d$, it is a Petrov-Galerkin projection. Only in case $P = I$, which means that all points are considered, Π is orthogonal and a Galerkin projection. In this event the projection is represented by $\Pi = U_d U_d^T$.

Note that the projection of missing point estimation can be seen as a catenation of two different projections: The first one projects onto the selected points and is given by $\Pi_P = P P^T$; the second one projects onto the POD subspace and is represented by $U_d U_d^T$. The requirement, that the catenation of the projections is itself a projection, necessitates the term $(U_d^T P P^T U_d)^{-1}$.

8.3 Point selection

In Section 8.1 it is mentioned that the selection matrix has to be carefully selected, since the quality of the reduced order model depends greatly on the point selection. In this section an approach for determining the points is outlined, which is applied in many publications on missing point estimation [5, 4, 64, 63]. It is based on minimizing the condition number of the matrix $C = (U_d^T P P^T U_d)$ and can be interpreted as preserving orthogonality of the matrix as well as possible. In the following, a motivation for this criterion shall be given.

Obviously, the points shall be chosen such that they contain as much information as possible. Note that the information of each point is reflected in the POD basis U_d. Hence, the matrix $C = (U_d^T P P^T U_d)$ gives an insight into the information in the point selection defined by the matrix P.

Note that in case all points are selected, then $C = I$ and in particular C is orthogonal. Furthermore, note that orthogonal matrices are perfectly conditioned matrices. Hence, by minimizing the condition number of C, indirectly the orthogonality of C is attempted to be preserved as well as possible.

In the following Lemma, the property used above, that orthogonal matrices are perfectly conditioned, shall be justified mathematically.

Lemma 8.1 *The condition number of a square matrix $C \in \mathbb{R}^{d \times d}$ is 1 if and only if C is a multiple of an orthogonal matrix.*

Proof: The condition number of a matrix is defined as the ratio of the largest and smallest singular values, i.e. $\kappa(C) = \frac{\sigma_1}{\sigma_d}$. Obviously, it holds

$$\kappa(C) = \frac{\sigma_1}{\sigma_d} = 1 \Longleftrightarrow \sigma_1 = \sigma_d. \tag{8.6}$$

Hence, if the condition number of C is 1, then the singular value matrix of C can be written as $\Sigma_C = \sigma_1 I$, since all singular values are equal to σ_1. The SVD of C thus yields

$$C = U_C \Sigma_C V_C^T = \sigma_1 U_C I V_C^T = \sigma_1 U_C V_C^T = \sigma_1 Q,$$

where Q is an orthogonal matrix, because it is the product of two orthogonal matrices.

On the other hand, if C is the multiple of an orthogonal matrix, that is $C = \sigma Q$, where σ is a scalar and Q an orthogonal matrix, then an SVD of $C = U_C \Sigma_C V_C^T$ can be constructed by setting $U_C = Q$, $\Sigma_C = \sigma I$, and $V_C = I$. Since the smallest singular value is equal to the largest one, it follows with (8.6) that the condition number of C is 1. ■

Selecting an optimal set of points is obviously a combinatorial problem. As these kinds of problems are very time-consuming, in [71] a greedy approach is proposed. Algorithm 2 sketches the idea of the greedy approach: Given the POD basis and an index set $\mathbb{I}$ of points[3], a subset $\mathbb{X}$ of $\mathbb{I}$ is determined by choosing in each iteration step the point, which minimizes the condition number of C. This is repeated until a user defined upper bound for the condition number is reached, or a certain number of points $\bar{n}$ has been chosen.

Algorithm 2 Point Selection Algorithm

```
Input: $\mathbb{I}$, $U_d$, $\delta$          # Indices of preselected points, POD basis, and user
                                               defined bound for condition number
Output: $\mathbb{X}$                         # Indices of selected points
$P = [\ ]$, $\mathbb{X} = \{\}$             # Initialize selection matrix $P$ and index set $\mathbb{X}$
$\kappa_{\min} = \infty$                     # Initialize minimal condition number
while $\kappa_{\min} > \delta$ and $|\mathbb{X}| \leq \bar{n}$   # Repeat until target condition number is met
    for $i \in \mathbb{I} \backslash \mathbb{X}$                 # Loop over points that have not been considered
        $\bar{P} = \begin{bmatrix} P & \boldsymbol{e}_i \end{bmatrix}$   # Add current index' unit vector to selection matrix
        $\kappa_i = \kappa(U_d^T \bar{P} \bar{P}^T U_d)$            # Compute condition number after adding index
    end for
    $j = \arg\min_i \kappa_i$,               # Determine optimal index
    $\kappa_{\min} = \kappa_j$               # Update minimal condition number $\kappa_{\min}$
    $\mathbb{X} = \mathbb{X} \cup \{j\}$,    # Update index set $\mathbb{X}$ with chosen index
    $P = \begin{bmatrix} P & \boldsymbol{e}_j \end{bmatrix}$    # Update selection matrix $P$
end while
```

---

[3] This index set could be the indices either of all points or of a subset of points, which seem to be most likely to be selected. By choosing a subset the algorithm obviously becomes more efficient.

8.4 Error estimation

In this section an error estimate for the reduced order MPE system shall be derived. Note that it is very similar to the one given in [59], since both MPE and the DEIM [13] are nonlinear model order reduction techniques, which use projections to obtain an efficiently evaluable right hand side for the reduced model. The difference between the two methods lies in the way the right hand side of the full order system is approximated. In the DEIM reduced model, a POD basis is used to interpolate the right hand side, while in MPE, it is projected in an efficient manner. This will obviously have some consequences for the error estimation.

As in [59], the least upper bound *logarithmic Lipschitz constants* with respect to an inner product $(\cdot,\cdot)$ is utilized. It is defined for a map $\boldsymbol{f} : \mathcal{W} \subset \mathbb{R}^N \to \mathbb{R}^N$ as

$$M[\boldsymbol{f}] := \sup_{\boldsymbol{u} \neq \boldsymbol{v}} \frac{(\boldsymbol{u}-\boldsymbol{v}, \boldsymbol{f}(\boldsymbol{u}) - \boldsymbol{f}(\boldsymbol{v}))}{||\boldsymbol{u}-\boldsymbol{v}||^2}. \tag{8.7}$$

This definition can be found in [58].

Theorem 8.1 *Let the error of the solution of the reduced order MPE system be given by* $\boldsymbol{\epsilon}(t) = \boldsymbol{w}(t) - \hat{\boldsymbol{w}}(t)$, *where* $\boldsymbol{w}(t)$ *is the full order and* $\hat{\boldsymbol{w}}(t)$ *the reduced order model solution for* $t \in [0,T]$. *That is,* $\boldsymbol{w}(t)$ *solves the system* $\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{w}(t))$ *with the initial condition* $\boldsymbol{w}(0) = \boldsymbol{w}_0$ *and* $\hat{\boldsymbol{w}}(t)$ *solves* $\frac{\mathrm{d}\hat{\boldsymbol{w}}(t)}{\mathrm{d}t} = \Pi \boldsymbol{f}(\hat{\boldsymbol{w}}(t))$ *with* $\hat{\boldsymbol{w}}(0) = U_d U_d^T \boldsymbol{w}_0$. *Let* Π *be defined as in (8.5). Furthermore, let the right hand side of the full order system* $\boldsymbol{f}$ *be Lipschitz continuous with the Lipschitz constant* $L_{\boldsymbol{f}}$ *and let* $M[\Pi \boldsymbol{f}]$ *be the logarithmic Lipschitz constant of* $\Pi \boldsymbol{f}$ *as in (8.7). Then the error* $\boldsymbol{\epsilon}(t)$ *can be estimated by*

$$\int_0^T ||\boldsymbol{\epsilon}(t)||^2 dt \leq C \left(\int_0^T ||\boldsymbol{w}(t) - U_d U_d^T \boldsymbol{w}(t)||^2 dt + \int_0^T ||\boldsymbol{f}(\boldsymbol{w}(t)) - U_d U_d^T \boldsymbol{f}(\boldsymbol{w}(t))||^2 dt \right),$$

where U_d *is the truncated POD basis and*

$$C := \max\{1 + 2c_M \beta^2 \,,\, 2c_M \gamma^2\},$$

$$\beta = L_{\boldsymbol{f}} ||U_d^T \Pi||, \quad \gamma = ||U_d^T \Pi \bar{U}||,$$

$$c_M = \begin{cases} \frac{1}{M[\Pi \boldsymbol{f}]}(e^{2M[\Pi \boldsymbol{f}]T} - 1), & M[\Pi \boldsymbol{f}] \neq 0, \\ 2T, & M[\Pi \boldsymbol{f}] = 0. \end{cases}$$

Proof: Denote the component of the error caused by approximating in the POD subspace by $\boldsymbol{\epsilon}_0(t)$ and the one within the POD subspace by $\boldsymbol{\epsilon}_1(t)$, that is,

$$\boldsymbol{\epsilon}_0(t) = \boldsymbol{w}(t) - U_d U_d^T \boldsymbol{w}(t)$$

and

$$\boldsymbol{\epsilon}_1(t) = U_d U_d^T \boldsymbol{w}(t) - \hat{\boldsymbol{w}}(t).$$

The overall error is then given by $\boldsymbol{\epsilon}(t) = \boldsymbol{\epsilon}_0(t) + \boldsymbol{\epsilon}_1(t)$.

Let $U_d^T \boldsymbol{w}(t) = \boldsymbol{a}(t)$, where $\boldsymbol{a}(t)$ are the coefficients of the projection of the exact solution onto the POD subspace. Furthermore, let $\bar{U}^T \boldsymbol{w}(t) = \hat{\boldsymbol{a}}(t)$, where $\bar{U}$ are the disregarded POD basis vectors and $\hat{\boldsymbol{a}}(t)$ are the corresponding coefficients for the solution of the full order system. As a result, it follows that

$$\boldsymbol{w}(t) = U_d\boldsymbol{a}(t) + \bar{U}\hat{\boldsymbol{a}}(t) \tag{8.8}$$

and that

$$U_dU_d^T\boldsymbol{w}(t) = U_d\boldsymbol{a}(t).$$

Hence,

$$\boldsymbol{\epsilon}_0(t) = \boldsymbol{w}(t) - U_dU_d^T\boldsymbol{w}(t) = U_d\boldsymbol{a}(t) + \bar{U}\hat{\boldsymbol{a}}(t) - U_d\boldsymbol{a}(t) = \bar{U}\hat{\boldsymbol{a}}(t).$$

Assume that the MPE solution can be written as

$$\hat{\boldsymbol{w}}(t) = U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)), \tag{8.9}$$

where $\tilde{\boldsymbol{a}}(t)$ is a perturbation from the optimal coefficients $\boldsymbol{a}(t)$. Thus,

$$\boldsymbol{\epsilon}_1(t) = U_dU_d^T\boldsymbol{w}(t) - \hat{\boldsymbol{w}}(t) = U_d\boldsymbol{a}(t) - U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)) = U_d\tilde{\boldsymbol{a}}(t).$$

Note that due to the orthogonality of the columns of U_d and $\bar{U}$, the errors $\boldsymbol{\epsilon}_1(t) = U_d\tilde{\boldsymbol{a}}(t)$ and $\boldsymbol{\epsilon}_0(t) = \bar{U}\bar{\boldsymbol{a}}(t)$ are orthogonal to each other.

Now consider the overall error $\boldsymbol{\epsilon}(t) = \boldsymbol{w}(t) - \hat{\boldsymbol{w}}(t)$. Differentiating this equation and premultiplying with U_d^T yields

$$U_d^T\frac{\mathrm{d}\boldsymbol{\epsilon}(t)}{\mathrm{d}t} = U_d^T\left[\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} - \frac{\mathrm{d}\hat{\boldsymbol{w}}(t)}{\mathrm{d}t}\right] = U_d^T[\boldsymbol{f}(\boldsymbol{w}(t)) - \Pi\boldsymbol{f}(\hat{\boldsymbol{w}}(t))].$$

Note that $U_d^T\boldsymbol{\epsilon}_1(t) = \tilde{\boldsymbol{a}}(t)$ and $U_d^T\boldsymbol{\epsilon}_0(t) = \mathbf{0}$. Since $\boldsymbol{\epsilon}(t) = \boldsymbol{\epsilon}_0(t) + \boldsymbol{\epsilon}_1(t)$, it holds that $U_d^T\boldsymbol{\epsilon}(t) = \tilde{\boldsymbol{a}}(t)$. After inserting (8.8) and (8.9) in the above equation we have

$$\frac{\mathrm{d}\tilde{\boldsymbol{a}}(t)}{\mathrm{d}t} = U_d^T[\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \Pi\boldsymbol{f}(U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)))].$$

Next, let $\boldsymbol{r}(t) = U_d^T[\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \Pi\boldsymbol{f}(U_d\boldsymbol{a}(t))]$, then

$$\frac{\mathrm{d}\tilde{\boldsymbol{a}}(t)}{\mathrm{d}t} = U_d^T\Pi[\boldsymbol{f}(U_d\boldsymbol{a}(t)) - \boldsymbol{f}(U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)))] + \boldsymbol{r}(t).$$

With

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}||U_d\tilde{\boldsymbol{a}}(t)||^2 &= \frac{\mathrm{d}}{\mathrm{d}t}\left[\tilde{\boldsymbol{a}}^T(t)U_d^TU_d\tilde{\boldsymbol{a}}(t)\right] \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\left[\tilde{\boldsymbol{a}}^T(t)\right]U_d^TU_d\tilde{\boldsymbol{a}}(t) + \tilde{\boldsymbol{a}}^T(t)U_d^TU_d\frac{\mathrm{d}}{\mathrm{d}t}\left[\tilde{\boldsymbol{a}}(t)\right] \\
&= 2\left(U_d\tilde{\boldsymbol{a}}(t), U_d\frac{\mathrm{d}\tilde{\boldsymbol{a}}(t)}{\mathrm{d}t}\right)
\end{aligned}$$

and $\frac{\mathrm{d}}{\mathrm{d}t}||U_d\tilde{\boldsymbol{a}}(t)||^2 = 2||U_d\tilde{\boldsymbol{a}}(t)||\,\frac{\mathrm{d}}{\mathrm{d}t}||U_d\tilde{\boldsymbol{a}}(t)||$, it follows that

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}||U_d\tilde{\boldsymbol{a}}(t)|| &= \frac{\left(U_d\tilde{\boldsymbol{a}}(t), U_d\frac{\mathrm{d}\tilde{\boldsymbol{a}}(t)}{\mathrm{d}t}\right)}{||U_d\tilde{\boldsymbol{a}}(t)||} \\
&= \frac{\left(U_d\tilde{\boldsymbol{a}}(t), U_dU_d^T\Pi[\boldsymbol{f}(U_d\boldsymbol{a}(t)) - \boldsymbol{f}(U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)))] + U_d\boldsymbol{r}(t)\right)}{||U_d\tilde{\boldsymbol{a}}(t)||} \\
&= \frac{(U_d\tilde{\boldsymbol{a}}(t), \Pi[\boldsymbol{f}(U_d\boldsymbol{a}(t)) - \boldsymbol{f}(U_d(\boldsymbol{a}(t) - \tilde{\boldsymbol{a}}(t)))])}{||U_d\tilde{\boldsymbol{a}}(t)||} + \frac{(U_d\tilde{\boldsymbol{a}}(t), U_d\,\boldsymbol{r}(t))}{||U_d\tilde{\boldsymbol{a}}(t)||} \\
&\leq M[\Pi\boldsymbol{f}]\,||U_d\tilde{\boldsymbol{a}}(t)|| + ||U_d\,\boldsymbol{r}(t)||
\end{aligned} \tag{8.10}$$

holds. Note that in the last step, the inequality is obtained by observing that the first term is bounded from above by the logarithmic Lipschitz constant defined in (8.7). The inequality for the second term follows from the Cauchy-Schwarz inequality $(U_d\tilde{\boldsymbol{a}}(t), U_d\,\boldsymbol{r}(t)) \leq ||U_d\,\boldsymbol{r}(t)||\cdot||U_d\tilde{\boldsymbol{a}}(t)||$.

Next, a solution to the differential inequality (8.10) will be given. To this end, subtract $M[\Pi\boldsymbol{f}]||U_d\tilde{\boldsymbol{a}}(t)||$ in (8.10) and subsequently multiply with $\mathrm{e}^{-M[\Pi\boldsymbol{f}]t} > 0$. This yields on the left hand side

$$\frac{\mathrm{d}}{\mathrm{d}t}(||U_d\tilde{\boldsymbol{a}}(t)||)\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t} - M[\Pi\boldsymbol{f}]||U_d\tilde{\boldsymbol{a}}(t)||\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t} \;=\; \frac{\mathrm{d}}{\mathrm{d}t}\left(||U_d\tilde{\boldsymbol{a}}(t)||\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t}\right),$$

where the above equality follows directly from the chain rule of differentiation. Hence, (8.10) altogether becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(||U_d\tilde{\boldsymbol{a}}(t)||\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t}\right) \;\leq\; ||U_d\,\boldsymbol{r}(t)||\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t}.$$

By integrating both sides of this inequality with respect to time

$$||U_d\tilde{\boldsymbol{a}}(t)||\,\mathrm{e}^{-M[\Pi\boldsymbol{f}]t} \leq \int_0^t \mathrm{e}^{-M[\Pi\boldsymbol{f}]\tau}\,||U_d\boldsymbol{r}(\tau)||d\tau + c \tag{8.11}$$

follows, where $c = \mathrm{e}^{M[\Pi\boldsymbol{f}]t}||U_d\tilde{\boldsymbol{a}}(0)||$ is a constant. Since the initial condition of the reduced system lies exactly in the POD subspace, that is, $\hat{\boldsymbol{w}}(0) = U_dU_d^T\boldsymbol{w}_0 = U_dU_d^T(U_d\boldsymbol{a}(0) - \bar{U}\bar{\boldsymbol{a}}(0)) = U_d\boldsymbol{a}(0)$, it holds that $\tilde{\boldsymbol{a}}(0) = \boldsymbol{0}$ and thus $c = 0$. In summary, a solution to the differential inequality (8.10) is given by

$$||\boldsymbol{\epsilon}_1(t)|| = ||U_d\tilde{\boldsymbol{a}}(t)|| \;\leq\; \int_0^t \mathrm{e}^{M[\Pi\boldsymbol{f}](t-\tau)}||U_d\boldsymbol{r}(\tau)||d\tau. \tag{8.12}$$

In the following an upper bound for $||U_d\boldsymbol{r}(t)||$ shall be derived. Note that

$$\begin{aligned}
\boldsymbol{r}(t) &= U_d^T[\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \Pi\boldsymbol{f}(U_d\boldsymbol{a}(t))] \\
&= U_d^T[(I - \Pi + \Pi)\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \Pi\boldsymbol{f}(U_d\boldsymbol{a}(t))] \\
&= U_d^T\Pi[\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \boldsymbol{f}(U_d\boldsymbol{a}(t))] + U_d^T(I - \Pi)\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)).
\end{aligned} \tag{8.13}$$

Decompose the right hand side of the full order system into $\boldsymbol{f}(\boldsymbol{w}(t)) = U_d\boldsymbol{b}(t) + \bar{U}\bar{\boldsymbol{b}}(t)$, where $\boldsymbol{b}(t)$ and $\bar{\boldsymbol{b}}(t)$ are suitable coefficients for representing the $\boldsymbol{f}$ in terms of the POD vectors and the disregarded POD vectors, respectively. Then

$$\begin{aligned}
U_d^T(I - \Pi)\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) &= U_d^T(I - \Pi)[U_d\boldsymbol{b}(t) + \bar{U}\bar{\boldsymbol{b}}(t)] \\
&= \boldsymbol{b}(t) + U_d^T\bar{U}\bar{\boldsymbol{b}}(t) - U_d^T\Pi\bar{U}\bar{\boldsymbol{b}}(t) \\
&\quad - U_d^TU_d(U_d^TPP^TU_d)^{-1}U_d^TPP^TU_d\boldsymbol{b}(t) \\
&= \boldsymbol{b}(t) - \boldsymbol{b}(t) - U_d^T\Pi\bar{U}\bar{\boldsymbol{b}}(t) \\
&= -U_d^T\Pi\bar{U}\bar{\boldsymbol{b}}(t).
\end{aligned}$$

This along with the Lipschitz continuity of $\boldsymbol{f}$, which implies that

$$||\boldsymbol{f}(U_d\boldsymbol{a}(t) + \bar{U}\bar{\boldsymbol{a}}(t)) - \boldsymbol{f}(U_d(\boldsymbol{a}(t)))|| \leq L_{\boldsymbol{f}}||\bar{U}\bar{\boldsymbol{a}}(t)|| = L_{\boldsymbol{f}}||\boldsymbol{\epsilon}_0(t)||,$$

yields

$$||\boldsymbol{r}(t)|| \leq \beta||\boldsymbol{\epsilon}_0(t)|| + \gamma||\bar{\boldsymbol{b}}(t)||,$$

where $\beta = L_{\boldsymbol{f}}||U_d^T\Pi||$ and $\gamma = ||U_d^T\Pi\bar{U}||$. Also note that $||U_d\boldsymbol{r}(t)||^2 = \boldsymbol{r}(t)^T U_d^T U_d \boldsymbol{r}(t) = \boldsymbol{r}(t)^T\boldsymbol{r}(t) = ||\boldsymbol{r}(t)||^2$.

By squaring both sides of inequality (8.12) and with the definition of

$$q_M(t) := \int_0^t \mathrm{e}^{2M[\Pi\boldsymbol{f}](t-\tau)}d\tau = \begin{cases} \frac{1}{2M[\Pi\boldsymbol{f}]}(\mathrm{e}^{2M[\Pi\boldsymbol{f}]t} - 1), & M[\Pi\boldsymbol{f}] \neq 0 \\ t, & M[\Pi\boldsymbol{f}] = 0, \end{cases}$$

the inequality

$$\begin{aligned} ||\boldsymbol{\epsilon}_1(t)||^2 &\leq \left(\int_0^t \mathrm{e}^{M[\Pi\boldsymbol{f}](t-\tau)}||U_d\boldsymbol{r}(\tau)||d\tau\right)^2 \leq \int_0^t \mathrm{e}^{2M[\Pi\boldsymbol{f}](t-\tau)}d\tau \int_0^t ||U_d\boldsymbol{r}(\tau)||^2 d\tau \\ &\leq q_M(t)\int_0^t ||U_d\boldsymbol{r}(\tau)||^2 d\tau \;\leq\; 2q_M(T)\left(\beta^2\int_0^T ||\boldsymbol{\epsilon}_0(t)||^2 dt + \gamma^2\int_0^T ||\bar{\boldsymbol{b}}(t)||^2 dt\right) \end{aligned}$$

is obtained. Note that in the second step the Cauchy-Schwarz inequality is used.

Since the errors $\boldsymbol{\epsilon}_0$ and $\boldsymbol{\epsilon}_1$ are orthogonal to each other, the Pythagorean theorem holds. Hence,

$$\begin{aligned} \int_0^T ||\boldsymbol{\epsilon}(t)||^2 dt &= \int_0^T ||\boldsymbol{\epsilon}_0(t)||^2 dt + \int_0^T ||\boldsymbol{\epsilon}_1(t)||^2 dt \\ &= \left[1 + 2q_M(T)\beta^2\right]\int_0^T ||\boldsymbol{\epsilon}_0(t)||^2 dt + 2q_M(T)\gamma^2\int_0^T ||\bar{\boldsymbol{b}}(t)||^2 dt \\ &\leq C\left(\int_0^T ||\boldsymbol{\epsilon}_0(t)||^2 dt + \int_0^T ||\bar{\boldsymbol{b}}(t)||^2 dt\right), \end{aligned}$$

where $C := \max\{1 + 2q_M(T)\beta^2\,,\, 2q_M(T)\gamma^2\}$. With $(I - U_dU_d^T)\boldsymbol{f}(\boldsymbol{w}(t)) = U_d\boldsymbol{b}(t) + \bar{U}\bar{\boldsymbol{b}}(t) - U_d\boldsymbol{b}(t) = \bar{U}\bar{\boldsymbol{b}}(t)$ and $||\bar{U}\bar{\boldsymbol{b}}(t)|| = ||\bar{\boldsymbol{b}}(t)||$ follows the result. ■

Note that Theorem 8.1 states that the error of the MPE reduced system can be bounded from above by a multiple of the sum of the projection error with respect to the POD subspace of the solutions $\boldsymbol{w}(t)$ and the projection error in the POD subspace of the right hand side $\boldsymbol{f}(t)$ of the system. If all points are considered in the MPE, that is $\Pi = U_dU_d^T$, it directly follows that $\gamma = ||U_d^T\Pi\bar{U}|| = 0$. Thus, the projection error of the right hand side $\boldsymbol{f}(t)$ can be neglected in the error estimation in case of Galerkin projection.

While Theorem 8.1 gives a theoretical insight into the error of MPE reduced order system, its error estimation is not suited in practice for fluid dynamical applications, since it is impossible to determine the Lipschitz constants $L_{\boldsymbol{f}}$ and $M[\Pi\boldsymbol{f}]$ for the right hand side $\boldsymbol{f}(t)$. As a matter of fact, $\boldsymbol{f}(t)$ might not be Lipschitz continuous at all.

But even without such an error estimator, MPE is applicable to practical applications such the simulation of aerodynamic flows. However, it cannot be guaranteed that the error is below a particular threshold. As a result, MPE has to be verified by numerical experiments for applications, where the Lipschitz constants are either unknown or do not exist. In particular, for transonic flows, for which shocks occur or – mathematically formulated – where discontinuities exist in the flow field, MPE should be used with caution, since this case the Lipschitz constants will not exist.

Part III

Model order reduction for steady aerodynamic applications

Chapter 9

Implementation of MPE

While in the previous chapters different techniques such as methods for solving the governing equations in CFD, proper orthogonal decomposition, and missing point estimation have been introduced in a general setting, the purpose of this chapter is to acquaint the reader with the actual implementation of the missing point estimation for steady applications in this work.

The chapter starts by introducing the full order model and the way the vector of conservative variables are collocated. The latter is important for the proper orthogonal decomposition, which is described next. Unlike in Chapter 7, POD is not computed with the Euclidean scalar product, but with a discrete version of the L_2 scalar product for discrete flow solutions. The same scalar product is used for the orthogonality conditions in the section on missing point estimation, in which the way that the points are selected for MPE is also addressed. In Section 9.4 the method used for solving the MPE reduced order model are described. The chapter finally closes with some remarks on the source code of the MPE implementation.

9.1 The full order model

In this work the full order model (FOM) are the spatially discretized governing equations (4.3) of fluid flows. Remember that this form of the equations is obtained by spatially discretizing the governing equations with a finite-volume approach. It reads

$$\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = -\tilde{\Omega}^{-1}\boldsymbol{R}(\boldsymbol{w}(t)) \tag{9.1}$$

and is a set of ordinary differential equations. In this thesis steady state problems are considered, for which the time derivative is (approximately) zero, i.e., $\mathbf{0} = -\tilde{\Omega}^{-1}\boldsymbol{R}(\boldsymbol{w}(t))$.

In case of the Euler equations $\boldsymbol{w}(t)$ stands for the discrete vector of conservative variables. For our purpose, it is most convenient if $\boldsymbol{w}(t)$ is collocated such that each variable appears blockwise, that is

$$\boldsymbol{w}^T = \left(\ \boldsymbol{\rho}^T,\ \ \boldsymbol{\rho v}_x^T,\ \ \boldsymbol{\rho v}_y^T,\ \ \boldsymbol{\rho v}_z^T,\ \ \boldsymbol{\rho E}^T\ \right) \in \mathbb{R}^{5N}, \tag{9.2}$$

where each of the entries $\boldsymbol{\rho}$, $\boldsymbol{\rho v_x}$, $\boldsymbol{\rho v_y}$, $\boldsymbol{\rho v_z}$, and $\boldsymbol{\rho E}$ is a vector of size N. The residual vector $\boldsymbol{R} \in \mathbb{R}^{5N}$ is ordered accordingly. It represents the discretization of the surface integral in (3.23).

In case of using the Spalart-Allmaras turbulence model, along with the Navier-Stokes equations the eddy-viscosity variable $\tilde{\nu}$ is appended to the conservative variables, i.e.

$$\boldsymbol{w}^T = \left(\boldsymbol{\rho}^T \ \boldsymbol{\rho v_x}^T \ \boldsymbol{\rho v_y}^T \ \boldsymbol{\rho v_z}^T \ \boldsymbol{\rho E}^T \ \tilde{\boldsymbol{\nu}}^T \right) \in \mathbb{R}^{6N}. \tag{9.3}$$

The corresponding residual vector $\boldsymbol{R} \in \mathbb{R}^{6N}$ is once again ordered accordingly.

Finally, $\tilde{\Omega} = \operatorname{diag}(\Omega, \ldots, \Omega) \in \mathbb{R}^{sN \times sN}$ with $\Omega \in \mathbb{R}^{N \times N}$ being a diagonal matrix, whose entries are the volumes of the cells Ω_i. Note that s denotes the number of variables. Thus for the Euler equations $s = 5$ and for the Navier-Stokes equations $s = 6$. In case the two-dimensional problem is considered, the variable $\boldsymbol{\rho v}_y$ usually drops out and therefore s is reduced by one.

9.2 POD for discretized flow solutions

In the following the proper orthogonal decomposition shall be described for discretized flow solutions of the form (9.2) or (9.3). Since POD has been derived for Hilbert spaces in Section 7.2, a scalar product for discretized flow solutions shall be defined such that results of Section 7.2 can be used.

Consider one flow quantity such as density and let $\upsilon, \nu \in L_2$ be the functions of that particular flow quantity of two flows at different flow conditions. Furthermore, consider the partitioning of the domain Ω into control volumes Ω_i, as it is done in cell-vertex schemes with dual control volumes (see Section 4.2). Then it is assumed that υ, ν are constant for each Ω_i with $i = 1, \ldots, N$, where N is the number of grid cells (control volumes). Let $\upsilon_i = \upsilon|_{\Omega_i}$ and $\nu_i = \nu|_{\Omega_i}$. Then the L_2 scalar product becomes

$$\begin{aligned} (\upsilon, \nu)_{L_2} &= \int_\Omega \upsilon \cdot \nu \, \mathrm{d}\Omega = \sum_{i=1}^N \int_{\Omega_i} \upsilon \cdot \nu \, \mathrm{d}\Omega_i \\ &= \sum_{i=1}^N \Omega_i \cdot \upsilon_i \cdot \nu_i = \boldsymbol{\upsilon}^T \Omega \boldsymbol{\nu} \end{aligned} \tag{9.4}$$

in discrete form, where Ω_i and Ω stand – with some abuse of notation – for the control volumes itself as well as for their volumes. Furthermore, $\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_N)$ and $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_N)$.

Now consider discrete flow solutions $\boldsymbol{w}_1, \boldsymbol{w}_2$ of the form (9.2) or (9.3). The scalar product of $\boldsymbol{w}_1, \boldsymbol{w}_2$ is then given by

$$(\boldsymbol{w}_1, \boldsymbol{w}_2)_{\tilde{\Omega}} = \boldsymbol{w}_1^T \tilde{\Omega} \boldsymbol{w}_2 \tag{9.5}$$

with $\tilde{\Omega} = \operatorname{diag}(\Omega, \ldots, \Omega) \in \mathbb{R}^{sN \times sN}$. This scalar product will be called the scalar product of discretized flow solutions.

With this definition the eigenvalue problem (7.4) of POD can be computed numerically. Assume that a finite number m of snapshots is given. Define the averaging operation to be $\mathrm{Avg} = \sum_{i=1}^m (.)$. Note that the factor $\frac{1}{m}$, typically used for computing the average, is dropped, since it is merely a scaling factor in the following computations. With these definitions, equation

(7.4) translates to

$$\begin{aligned}\lambda \boldsymbol{u} &= \sum_{k=1}^{m} (\boldsymbol{u}, \boldsymbol{w}_k)_{\tilde{\Omega}} \, \boldsymbol{w}_k \\ &= YY^T \tilde{\Omega} \boldsymbol{u} \end{aligned} \tag{9.6}$$

in the discrete case subject to $||\boldsymbol{u}||^2 = \boldsymbol{u}^T \tilde{\Omega} \boldsymbol{u} = 1$, where Y is the (shifted) snapshot matrix.

Premultiplying the above equation with $\tilde{\Omega}^{\frac{1}{2}}$ yields the equivalent Euclidean eigenvalue problem

$$\tilde{\Omega}^{\frac{1}{2}} YY^T \tilde{\Omega}^{\frac{1}{2}} \tilde{\boldsymbol{u}} = \lambda \tilde{\boldsymbol{u}} \tag{9.7}$$

subject to $||\tilde{\boldsymbol{u}}||_2^2 = \tilde{\boldsymbol{u}}^T \tilde{\boldsymbol{u}} = 1$, where $\tilde{\boldsymbol{u}} = \tilde{\Omega}^{\frac{1}{2}} \boldsymbol{u}$. The solution to (9.7) is closely related to the singular value decomposition (SVD) of the matrix $\tilde{\Omega}^{\frac{1}{2}} Y$ (see [67]), which according to Theorem 5.1 exists for any real matrix. As seen in Section 7.1 the left singular vectors are the solutions to (9.7). Thus, the POD problem with the discrete L_2 scalar product for discrete flow solutions (9.6) is solved by the vectors $\boldsymbol{u} = \tilde{\Omega}^{-\frac{1}{2}} \tilde{\boldsymbol{u}}$.

Note that eigenvalue problem (9.7) is of size sN, which is the number of grid cells times the number of variables and is typically very large. Especially, for three-dimensional wing-body configurations solving the problem becomes infeasible. For this reason, in [57] Sirovich suggested to solve the eigenvalue problem (5.3) instead of (5.2). In our context this problem translates to

$$Y^T \tilde{\Omega} Y \boldsymbol{v} = \lambda \boldsymbol{v}, \tag{9.8}$$

which is of size m. This is called the method of snapshots. Note that $\boldsymbol{v}$ are the right singular vectors of the matrix $\tilde{\Omega}^{\frac{1}{2}} Y$. The eigenvectors to (9.6), which are the left singular vectors of $\tilde{\Omega}^{\frac{1}{2}} Y$, can then be obtained via

$$\boldsymbol{u}_i = \tilde{\Omega}^{-\frac{1}{2}} \tilde{\boldsymbol{u}}_i = \frac{1}{\sqrt{\lambda_i}} Y \boldsymbol{v}_i .$$

Note that the affine POD representation using the average shift (refer to Section 7.3) is employed in all numerical experiments in this work. Hence, the snapshot matrix is given by $Y = [(\boldsymbol{w}_1 - \bar{\boldsymbol{w}}) \ \ldots \ (\boldsymbol{w}_M - \bar{\boldsymbol{w}})] \in \mathbb{R}^{sN \times m}$ and solutions are represented as $\boldsymbol{w} = U_d \, \boldsymbol{a}(t) + \bar{\boldsymbol{w}}$.

To sum up, the above POD methodology is summarized in Algorithm 3.

Algorithm 3 Implementation of POD

```
Input: Snapshots $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m$
Output: POD basis vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d$
1: Compute the average of the snapshots $\bar{\boldsymbol{w}} = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{w}_i$.
2: Set up the snapshot matrix $Y = [(\boldsymbol{w}_1 - \bar{\boldsymbol{w}}) \ \ldots \ (\boldsymbol{w}_M - \bar{\boldsymbol{w}})] \in \mathbb{R}^{sN \times m}$.
3: Compute the eigenvalue problem $Y^T \tilde{\Omega} Y \boldsymbol{v} = \lambda \boldsymbol{v}$.
4: Determine the number $d$ of relevant POD basis vectors.
5: Compute the POD vectors by $\boldsymbol{u}_i = \tilde{\Omega}^{-\frac{1}{2}} \tilde{\boldsymbol{u}}_i = \frac{1}{\sqrt{\lambda_i}} Y \boldsymbol{v}_i$ for $i = 1, \ldots, d$.
```

9.3 MPE for discretized flow solutions

In the following the missing point estimation will be reviewed in context of the governing equations of computational fluid dynamics and with respect to the scalar product introduced in the previous section.

Consider the full order model (9.1) with the POD representation of the form (7.7) inserted for the discrete flow $\boldsymbol{w}$, that is

$$U_d \frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} = \tilde{\Omega}^{-1} \boldsymbol{R}\left(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}\right) + \boldsymbol{\epsilon}_0. \tag{9.9}$$

Remember that $\boldsymbol{\epsilon}_0$ is the error which is introduced by this approximation and that the shift is time-independent, i.e. $\frac{\mathrm{d}\bar{\boldsymbol{w}}}{\mathrm{d}t} = \mathbf{0}$. In this work the shift $\bar{\boldsymbol{w}}$ will be the average of the snapshots as in (7.8).

The idea of missing point estimation is to construct a reduced order model such that the residual $\boldsymbol{R}$ has to be evaluated only at a subset of all the control volumes of the computational mesh. The indices of the points of this subset shall be given by the indices $\mathbb{X} = \{j_1, \ldots, j_n\} \subset \{1, \ldots, N\}$.

While in Section 8.1 the MPE is derived for a set of equations with only one (continuous) variable, in context of CFD there are s flow variables. Hence, the selection matrix has to be defined accordingly. It is given by $\tilde{P} = \operatorname{diag}(P, \ldots, P) \in \mathbb{R}^{sN \times sn}$, where

$$P = \begin{pmatrix} \boldsymbol{e}_{j_1} & \cdots & \boldsymbol{e}_{j_n} \end{pmatrix} \in \mathbb{R}^{N \times n},$$

is defined as before with the jth unit vector $\boldsymbol{e}_j \in \mathbb{R}^N$. A projection onto the chosen points $\mathbb{X} = \{j_1, \ldots, j_n\}$ for all variables is then given by $\Pi_{\tilde{P}} = \tilde{P}\tilde{P}^T$. After applying this projection, equation (9.9) reads

$$\tilde{P}\tilde{P}^T U_d \frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} - \tilde{P}\tilde{P}^T \tilde{\Omega}^{-1} \boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}) - \tilde{P}\tilde{P}^T \boldsymbol{\epsilon}_0 = 0. \tag{9.10}$$

Recall that the next step is to impose orthogonality conditions upon the system, which state that the left hand side of (9.10) is orthogonal to each POD basis vector. Obviously, orthogonality depends on the scalar product. In order to be consistent, the scalar product of discretized flows as defined in (9.5) is used. Thus

$$\left(\boldsymbol{u}_j \,, \tilde{P}\tilde{P}^T U_d \frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} - \tilde{P}\tilde{P}^T \tilde{\Omega}^{-1} \boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})\right)_{\tilde{\Omega}} = 0, \qquad j = 1, \ldots, d \tag{9.11}$$

or in matrix notation

$$U_d^T \tilde{\Omega} \left(\tilde{P}\tilde{P}^T U_d \frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} - \tilde{P}\tilde{P}^T \tilde{\Omega}^{-1} \boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})\right) = 0. \tag{9.12}$$

Note that since both $\tilde{P}\tilde{P}^T$ and $\tilde{\Omega}$ are diagonal matrices, their product is commutative. Consequently, $U_d^T \tilde{\Omega} \tilde{P}\tilde{P}^T \tilde{\Omega}^{-1} = U_d^T \tilde{P}\tilde{P}^T$. By premultiplying equation (9.12) with $(U_d^T \tilde{\Omega} \tilde{P}\tilde{P}^T U_d)^{-1}$, the MPE system is obtained and reads

$$\frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} = (U_d^T \tilde{\Omega} \tilde{P}\tilde{P}^T U_d)^{-1} U_d^T \tilde{P}\tilde{P}^T \boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}}). \tag{9.13}$$

Note that this reduced order system consists only of $d \ll sN$ equations and thus is of small complexity.

For an adequate choice of the selection matrix P, the inverse $(U_d^T \tilde{\Omega} \tilde{P} \tilde{P}^T U_d)^{-1}$ exists. The point selection shall be addressed in the following subsection.

9.3.1 Point selection

Obviously, the point selection plays an important role in the performance of the missing point estimation. It has an influence both on the accuracy of the solution as well as the time efficiency of the computation.

Choosing point selections based on Algorithm 2 in Section 8.3 is far too costly for the application considered in this work. This is due to the fact that the number of grid cells is too large and that the point selection cannot be considered as an offline cost like in other applications. For example, for flow control [71] the time spent for creating the model is irrelevant. What matters, is the time spent when the flow control is active. In our case, the goal is not to obtain the most efficient reduced order model irrespective of the cost for constructing it. Rather, the goal is to reduce the time spent on obtaining a solution at a single or at a set of parameter settings compared to a full order CFD computation. Note that this includes the construction of the reduced order model as well.

As a result, the points shall be selected without significant cost. For this reason they are chosen based on their distance to the surface. In fact, the point selections which are considered for two-dimensional Navier-Stokes test cases in Chapter 10 are: some farfield points[1] , all farfield points, all points and finally, additional to the farfield, either points in the boundary layer or close to boundary layer. Specifics for the test cases will be given in Chapter 10.

For the Euler test cases in Chapter 10, the point selections are: all farfield points, points close to the surface, points further away from the surface and finally all points.

Note that in all point selections the farfield points are included. This is due to the fact that in Chapter 10 the angle of attack is the only system parameter. It enters the governing equations on the farfield boundary. In fact, as explained in Section 4.4, the velocities at the farfield boundary are set to their freestream values, which in turn depend on the angle of attack (see Equation (4.9)). Due to this dependence of the farfield points on the parameter, the farfield is always considered.

9.4 Solving the reduced order MPE model

In Subsection 4.3.2 it is outlined that there are two types of schemes, which can be used in order to solve a system of ordinary differential equations: Either explicit or implicit schemes can be employed. The difference between them lies in the time step, at which the residual

---

[1] Those points lying on the farfield boundary shall be referred to as *farfield points* in this work. In Figure 10.1(b) such a farfield point is visualized by a black dot.

is evaluated. Explicit schemes make use of the residual at previous time steps only, whereas implicit schemes also utilize the residual at the next time step.

Recall that in context of solving the full order model, implicit schemes allow greater time steps without loss of stability and are more robust. Explicit schemes, on the other hand, have the advantage that they are computationally more efficient per time step. For the reduced order MPE model, this advantage becomes less significant, since along with the order the computational effort is reduced.

Consider the implicit time-scheme

$$\frac{\Delta \boldsymbol{w}(t_n)}{\Delta t} = -\beta \cdot \Omega^{-1}\boldsymbol{R}(\boldsymbol{w}(t_{n+1})) - (1-\beta)\cdot \Omega^{-1}\boldsymbol{R}(\boldsymbol{w}(t_n)),$$

as in equation (4.5), but where the mass matrix M has been replaced by the identity matrix, since steady states are to be computed and thus temporal accuracy is not an issue. Again, β is a parameter, which can be chosen, and $\Delta \boldsymbol{w}(t_n) = \boldsymbol{w}(t_{n+1}) - \boldsymbol{w}(t_n)$ as well as $\Delta t = t_{n+1} - t_n$.

For the reduced order model this time-scheme reads

$$W_d^T \cdot \left[U_d \frac{\Delta\, \boldsymbol{a}(t_n)}{\Delta t} + \beta \cdot \Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_{n+1}) + \bar{\boldsymbol{w}}\right) + (1-\beta)\cdot \Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_n) + \bar{\boldsymbol{w}}\right)\right] = 0, \quad (9.14)$$

where $W_d^T = (U_d^T\tilde{\Omega}\tilde{P}\tilde{P}^T U_d)^{-1}U_d^T\Omega\tilde{P}\tilde{P}^T$ and $\Delta\boldsymbol{a}(t_n) = \boldsymbol{a}(t_{n+1}) - \boldsymbol{a}(t_n)$.

Since the POD coefficients $\boldsymbol{a}(t_{n+1})$ and thus also $\boldsymbol{R}\left(U_d\boldsymbol{a}(t_{n+1}) + \bar{\boldsymbol{w}}\right)$ are unknown in equation (9.14), the term $W_d^T\Omega^{-1}\boldsymbol{R}\left(U\boldsymbol{a}(t_{n+1})\right)$ is linearized by an approximate Taylor series given by

$$W_d^T\Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_{n+1}) + \bar{\boldsymbol{w}}\right) \approx W_d^T\Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_n) + \bar{\boldsymbol{w}}\right) + \frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial(U_d\boldsymbol{a} + \bar{\boldsymbol{w}})} U_d \Delta\, \boldsymbol{a}(t_n).$$

Note that

$$\frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial\boldsymbol{a}} = \frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial(U_d\boldsymbol{a} + \bar{\boldsymbol{w}})} \cdot \frac{\partial(U_d\boldsymbol{a} + \bar{\boldsymbol{w}})}{\partial\boldsymbol{a}} = \frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial(U_d\boldsymbol{a} + \bar{\boldsymbol{w}})} \cdot U_d.$$

Hence, equation (9.14) becomes

$$\left[\frac{W_d^T U_d}{\Delta t} + \beta\frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial\boldsymbol{a}}\right]\Delta\,\boldsymbol{a}(t_n) = -\,W_d^T\Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_n) + \bar{\boldsymbol{w}}\right).$$

As in Subsection 4.3.2, Newton's method is obtained by setting β to 1 and $\Delta t \to \infty$:

$$\frac{\partial\left(W_d^T\Omega^{-1}\boldsymbol{R}\right)}{\partial\boldsymbol{a}}\Delta\,\boldsymbol{a}(t_n) = -\,W_d^T\Omega^{-1}\boldsymbol{R}\left(U_d\boldsymbol{a}(t_n) + \bar{\boldsymbol{w}}\right). \quad (9.15)$$

Note that the above linear system (9.15) is of order d, which is equal to the dimension of the POD subspace. Since this order is typically a lot smaller than the order of the full order model, solving the above system is comparably inexpensive. Note that in this work the dimension d is typically smaller than ten.

In this thesis instead of using Newton's method, a related, but more refined approach called Powell's Dog Leg method [46, 42] is employed. This is due to the fact that the method can fail if the iterated solution is not a good estimation of the actual solution. For this reason Powell's Dog Leg method interpolates between the solution of the Newton and another method called the steepest descent approach. This is done in such a way that if the solution of the Newton method cannot be trusted, then the interpolation is biased toward the steepest descent direction; else the interpolation favors the solution of the Newton method. For details the reader is referred to [46].

In the end, an overall overview of the implementation of the missing point estimation is given in Algorithm 4.

Algorithm 4 Implementation of MPE

```
Compute snapshots, in case they are not available.
Determine POD basis with the help of Algorithm 3.
Select the points according to Subsection 9.3.1.
Set up the matrix factor $W_d^T = (U_d^T \tilde{\Omega} \tilde{P} \tilde{P}^T U_d)^{-1} U_d^T \Omega \tilde{P} \tilde{P}^T$ of the projection matrix $\Pi = U_d W_d^T$.
Solve the reduced system $\frac{\mathrm{d}\boldsymbol{a}}{\mathrm{d}t} = W_d^T \boldsymbol{R}(U_d \boldsymbol{a})$ with the help of Powell's Dog Leg method as described in Section 9.4.
```

9.5 Code description

After the in-depth description of the implementation of the numerical methods, focus shall now be given to the source code. For the various disciplines of the code, different numerical software is used, which will be described first before commenting on how these code segments are glued together.

As the developed software is intended to be used by the industrial partners of the ComFliTe project, a highly sophisticated CFD code capable of coping with industrial applications is used. In fact, the Tau code [22, 21], developed at the German Aerospace Center (DLR), is employed. The Tau code is a finite volume CFD solver, which employs many different spatial discretizations as well as turbulence models [28]. In this work it is not only used for computing the snapshots, but also for evaluating the residuals when solving the reduced order MPE system. Using the same code – or in other words, the same spatial discretization and turbulence model – in the full and the reduced order system is mandatory for comparing the quality of the results of the two models. This is due to the fact that both the spatial discretization scheme and the turbulence model have a great influence on the accuracy of the solution.

Note, however, that the sparse evaluation of the residual needed for the MPE is not yet implemented in Tau. Therefore the whole residual will be computed, but as a matter of fact, only those components actually needed for the reduced order model are used. Obviously, this is not efficient, but the purpose of this work is to show that the idea of missing point estimation works even for complex flow problems.

The computation of the POD basis is carried out by routines of a toolbox, called Smart [74], which is short for Surrogate Modeling for AeRo data Toolbox. The toolbox is written in C and

– in addition to computing the POD – is comprised of routines for reading snapshots, writing the solution files and different reduced order modelling techniques.

The reduced order model is solved with the help of Powell's Dog Leg method as described in Section 9.4. In particular, the realization of the Dog Leg method in the function *fsolve* of the subpackage *optimize* of *scipy* [29] is utilized. It is a wrapper around Minpack's *hybrd* and *hybrj* algorithms [44], which are written in Fortran. In our case only the function *hybrd* is used, which means that the Jacobian is evaluated by a finite difference approximation[2]. As a start vector for the algorithm, the average of the snapshots is chosen, that is $\boldsymbol{a} = \boldsymbol{0}$.

After the different code segments have been illustrated, the way that they are glued together is presented. Since the flow solver Tau – written in C – has a python interface, the functions of Tau can easily be called from a python script. For this reason python is chosen as a top level programming language. Because also *fsolve* can directly be used in python, only the needed Smart functions have to be wrapped to python. This is done with Swig, which is a software tool for automatically generating wrapping functions for scripting languages to call C code [8].

Figure 9.1 is a flow chart of the overall MPE code. Note that a white background stands for functions of the Smart toolbox, while black and gray represent Tau and the *fsolve* function of *scipy*, respectively. At first, the snapshots are read from files and the proper orthogonal decomposition is computed. With the thus obtained POD basis, the initial condition can be determined. In this work, this is done by inserting $\boldsymbol{a} = \boldsymbol{0}$ into the affine POD representation (7.7). That is, the average of the snapshots $\bar{\boldsymbol{w}}$ is chosen as a start solution. In order to compute the Jacobian, finite differences are employed. To this end, the initial flow as well as the other flows, defined by $U_d(\boldsymbol{a}+\Delta\boldsymbol{e}_i)+\bar{\boldsymbol{w}}$ with the step size Δ, are handed over to Tau, which computes the corresponding residuals $\boldsymbol{R}$. With the residuals $\boldsymbol{R}$, the reduced order residuals $W_d^T\boldsymbol{\Omega}^{-1}R$ are determined and each column of the Jacobian $J = \frac{\partial(W_d^T\Omega^{-1}\boldsymbol{R})}{\partial\boldsymbol{a}}$ is approximated by the finite differences $J_i = \frac{W_d^T\Omega^{-1}\boldsymbol{R}(U_d(\boldsymbol{a}+\Delta\boldsymbol{e}_i)+\bar{\boldsymbol{w}})-W_d^T\Omega^{-1}\boldsymbol{R}(U_d\boldsymbol{a}+\bar{\boldsymbol{w}})}{\Delta}$. As a next step, the first iteration of Powell's Dog Leg method is computed, yielding the coefficient vector $\boldsymbol{a}$. With this a solution is determined by the POD representation (7.7), the residual is computed by Tau and subsequently, the code checks, if the resulting reduced order residual $W_d^T\boldsymbol{\Omega}^{-1}R$ for the newly obtained solution is close enough to zero. If not, another iteration is carried out. This is repeated until convergence is detected. In the end the final solution is written.

In the flow chart as well as in the above paragraph, it becomes obvious that the *fsolve* function of *scipy*, the Tau code, and functions of the Smart toolbox had to be intertwined, which posed a challenge from a programming point of view.

---

[2] The ith column of the Jacobian is computed by $J_i = \frac{W_d^T\Omega^{-1}\boldsymbol{R}(U_d(\boldsymbol{a}+\Delta\boldsymbol{e}_i)+\bar{\boldsymbol{w}})-W_d^T\Omega^{-1}\boldsymbol{R}(U_d\boldsymbol{a}+\bar{\boldsymbol{w}})}{\Delta}$, where $\Delta = \sqrt{10^{-5}}$.

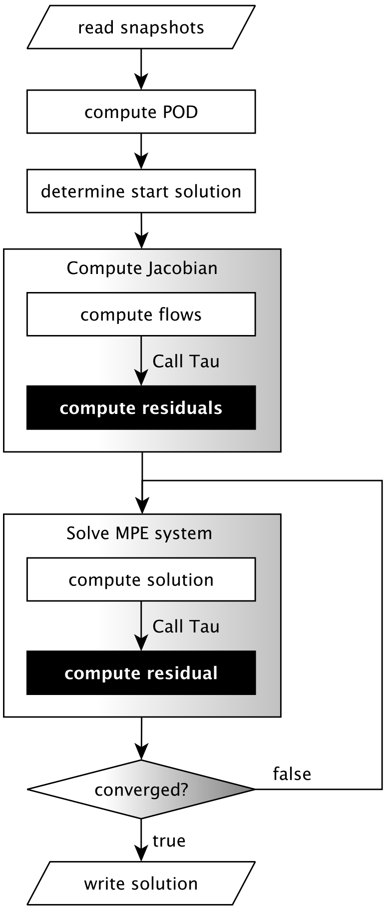


Figure 9.1: *Flow chart describing the source code of the implementation of the MPE model.*

Chapter 10

Numerical results for subsonic flows with the angle of attack as a system parameter

In this chapter numerical results for missing point estimation are shown for different test cases. It shall be shown that the MPE reduced order model (9.13) yields accurate results very rapidly. In Section 9.4 it has been outlined how this system is solved numerically. Besides, showing the good accuracy and efficiency of the method, the influence of most of the settings of the system such as, for example, the influence of the selection matrix and the number of modes being used, are investigated.

This is done for five different test cases. In each, the flow field for an angle of attack, which is not included in the snapshot set, is computed with the MPE reduced order model. In this chapter the Mach number is not a parameter. However, it will be in Chapter 11.

The first three test cases are representative of a particular class of problems. This class of problems comprises the simulation of subsonic flows, where in the MPE reduced order system only the angle of attack α is a parameter to the system and the considered angles of attack are below the static stall region.

In the first two test cases, the viscous flow around a two-element airfoil of the DLR-F15 configuration, is modeled. The first considers the DLR-F15 in take-off and the latter in landing conditions. For both tests, the considered geometries are two-dimensional section cuts of the wing and the flow is modeled with the Navier-Stokes equations.

In the third test case the inviscid flow field of a three-dimensional wing-body configuration, the DLR-F12, is simulated. Here, the Euler equations are used to compute the flow. This is done to keep the number of grid points small enough such that the solution is computable on a desktop computer. Note that the Euler equations typically require much less grid points than the Navier-Stokes equations, since the boundary layer does not have to be simulated.

The fourth test case considers the DLR-F15 in take-off conditions for higher angles of attack, which lie in the static stall region.

And finally, in the last test case, a transonic flow around the NACA0012 airfoil is considered.

Most of the snapshots for the test cases have been provided by Michael Mifsud, Ralf Zimmermann and Stefan Görtz of the German Aerospace Center.

10.1 Two-dimensional DLR-F15 airfoil in take-off conditions

As a first test case, the viscous flow past a two-element section cut of the wing of the DLR-F15 configuration at a freestream Mach number of $M_\infty = 0.22$, a Reynolds number of 10.549 million and a reference temperature of $118\,K$ is considered. Given a set of snapshots, which in this test case are steady state solutions for the angles of attack $\alpha \in \{0^\circ, 2^\circ, 4^\circ, 6^\circ, 8^\circ\}$, our goal is to compute the solution at $\alpha = 7^\circ$ at the same Mach number with the help of missing point estimation.

The snapshots are computed with the flow solver TAU [22, 21] by solving the Navier-Stokes equations using the Spalart-Allmaras one-equation turbulence model [10, Subsection 7.2.1]. The computation of the first snapshot at the lowest angle of attack ($\alpha = 0^\circ$) is initialized with freestream conditions, while the others are computed using the snapshot with the nearest angle of attack (two degrees below the current α) as a start solution. This procedure is followed in order to avoid – as much as possible – multiple values of the aerodynamic coefficients at a particular angle of attack due to aerodynamic hysteresis. In fact, it has been observed that the lift and drag coefficient is different, if the CFD computation is started from freestream values and from a solution at a lower angle of attack [41]. This phenomenon is called hysteresis.

Convergence to steady state is detected based on the absolute change in the aerodynamic coefficients. More precisely, if the change in the aerodynamic coefficients is below a user-defined tolerance over some specified interval of iterations, then the solution is considered to be converged. In this test case the interval is chosen to be 100 iterations and the tolerances for the change in the lift c_l, drag c_d and pitching moment c_m coefficient are set to 10^{-3}, 10^{-5}, and 0.1, respectively. Thus, e.g. for the lift coefficient, convergence is detected if $|c_l^k - c_l^{k-j}| < 10^{-3}$ for all $j = 1, \ldots, 100$, where k is the current number of iterations. Note that although the residual has not been used to detect convergence, it has been checked that the residual is sufficiently small.

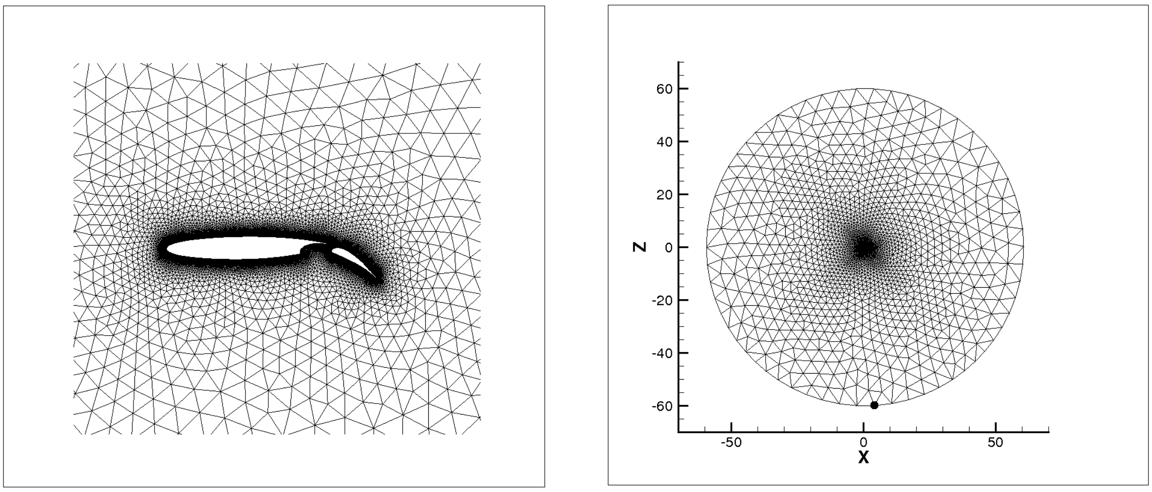
All computations are carried out on a hybrid grid, which is structured near the surface of the airfoil and unstructured otherwise. It consists of 50,809 grid points (control volumes) and is depicted in Figure 10.1.

Influence of the number of used modes: At first, compute the POD and take a look at the relative energy contained in each mode in the L_2 sense, which is given by

$$E(j) = \frac{\sigma_j^2}{\sum_{i=1}^{M} \sigma_i^2}, \tag{10.1}$$

where σ_i are the singular values of the POD.

Note that the relative energy content of the last mode is in fact exactly zero. This follows from Lemma 7.1. Numerically, instead of zero we obtain a number in the order of machine precision.



(a) Detailed view of the computational grid close to the surface.

(b) Full view with one marked point constituting point selection I of Table 10.4.

Figure 10.1: *The hybrid grid of the wing-flap configuration with* $N = 50,809$ *points.*

j	1	2	3	4	5
$E(j)$	$9.99571 \cdot 10^{-1}$	$4.27176 \cdot 10^{-4}$	$1.30249 \cdot 10^{-6}$	$1.08424 \cdot 10^{-7}$	0

Table 10.1: *Relative energy content of the modes corresponding to the snapshots at* $\alpha \in \{0°, 2°, 4°, 6°, 8°\}$ *of the high-lift take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*).*

Furthermore, note that the first mode contains more than 99.9 % of the energy. This is due to the L_2 scalar product as described in [72]: The most significant differences in the snapshots occur close to the surface. Since the L_2 inner product gives equal weighting to all regions of the computational domain, the region close to the surface, where these dramatic differences occur, makes up only a small part of the overall computational domain. As a result the snapshots are almost linearly dependent in the L_2 scalar product.

Since the relative energy content formally suggests that only one mode is enough to get a good representation of the solution, we want to investigate the impact of the truncation level of the POD modes further. For this purpose we use different numbers of modes d to simulate the solution at an angle of attack of $\alpha = 7°$ with the MPE. In order to reduce the influence of the selected points, we will use the same points for all simulations. In fact, we will restrict the residual evaluations to the farfield points.

Table 10.2 shows the aerodynamic coefficients as well as the associated errors with respect to the CFD reference solution for the different truncation levels. Note that the error in the drag coefficient c_d decreases with the number of modes. However, there is only a slight improvement when using more than two modes. The error of the lift coefficient c_l is almost constant and is least, when all modes are considered.

In order to investigate these results further, we look at the energy modes, which are displayed in Figure 10.2. Note that the values of the modes are not of physical, but of pure numerical

# of modes	c_l (error in %)	c_d (error in %)
$d = 1$	2.698 (0.48%)	$5.035 \cdot 10^{-2}$ (3.90%)
$d = 2$	2.699 (0.52%)	$4.891 \cdot 10^{-2}$ (0.93%)
$d = 3$	2.699 (0.52%)	$4.891 \cdot 10^{-2}$ (0.93%)
$d = 4$	2.696 (0.41%)	$4.865 \cdot 10^{-2}$ (0.39%)
CFD	2.685	$4.846 \cdot 10^{-2}$

Table 10.2: *Lift and drag coefficients of the take-off configuration (Re = 10.5 million, $M_\infty = 0.22$) computed with MPE, where the residual evaluations are restricted to the farfield points. The aerodynamic coefficients are predicted at $\alpha = 7°$ for different numbers of modes d and compared to the CFD reference solution at the same angle.*

# of modes	1	2	3	4
$d = 1$	2.39941			
$d = 2$	2.39940	$-7.04252 \cdot 10^{-3}$		
$d = 3$	2.39943	$-7.04524 \cdot 10^{-3}$	$3.26427 \cdot 10^{-4}$	
$d = 4$	2.39955	$-6.99386 \cdot 10^{-3}$	$2.95538 \cdot 10^{-4}$	$8.73545 \cdot 10^{-4}$

Table 10.3: *The POD coefficients for the modes obtained with MPE, where the residual evaluations are restricted to the farfield points, using different numbers of modes d.*

nature, since the average is subtracted and the data is non-dimensionalized. Each figure has its own scaling in order to better highlight its specific features. Also note that the scaling in the overall and the zoomed view of a particular mode differs as well.

Since the modes are multiplied with the POD coefficients to obtain the ROM solution, the POD coefficients are shown in Table 10.3. Note that the first mode coefficient is of much higher magnitude than all other coefficients. Furthermore, it is about the same, no matter how many modes are used.

When using two and three modes, the second coefficient is also almost the same. This – along with the fact that the third coefficient as well as the values of the third mode close to the surface are of smaller magnitude than those of the second mode – explains why the aerodynamic coefficients in Table 10.2 are the same.

Investigating the modes also gives an explanation why the error in the lift coefficient is slightly increased when using two rather than just one mode: Note that using the second mode significantly reduces the drag and thus also the error in this aerodynamic coefficient. It can be seen in Figure 10.2(d) that the second energy mode is positive around the leading edge as well as on the upper side of the airfoil. Note that the second mode is multiplied by a negative coefficient. As a result the energy becomes smaller around the leading edge and on the upper side of the airfoil when using the second mode. Since the energy is related to the pressure via the equation of state (3.17), the pressure also becomes smaller around the leading edge – reducing the drag – and on the upper side of the airfoil – increasing the lift – when using the second mode.

Since the prediction of the aerodynamic coefficient is best when choosing all modes, we will consider all of them from now on. Note also that even if all modes are chosen, the reduction of the order of the equations is dramatic nonetheless, since a system of order $5 \cdot N = 254,045$ has been reduced to a system of order $d = 4$.

If MPE is used to reconstruct the snapshots, the aerodynamic coefficients of the approximated flows with MPE using all four modes closely match those of the snapshots (computed with CFD). In fact, the error in the lift coefficient is of the order 10^{-3} and for the drag coefficient, it is of the order 10^{-5}. In general, numerical experience have shown that the reconstruction of the snapshots – no matter how many modes are used – are approximated better than angles of attack, which are not included in the snapshot set. Of course, this is expected.

Influence of the selection matrix: In the following we want to test, which impact the selection of the points has on the quality of the solution of the missing point estimation. To this end, we consider six different point selections:

Note that the farfield points are closely connected to the angle of attack. As a matter of fact, if no farfield point is included in the point selection, inaccurate results are obtained. In Point Selection II we will choose all 60 of them, while in Point Selection I we choose only a single farfield point that is marked in Figure 10.1(b). Note that at $\alpha = 0°$ the onflow in this point is parallel to the tangent of the computational domain. When the angle of attack is increased, the flow will enter the control volume more and more directly. For this reason the angle of attack has significant influence on the point and is thus considered. The importance of the point can also be observed in the POD energy modes in Figure 10.2. In the point the values of the modes are minimal or maximal compared to all other farfield points.

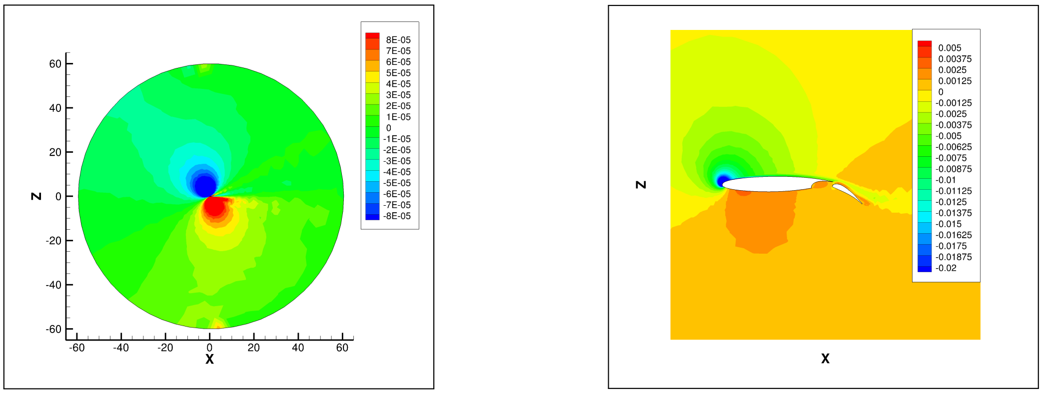
In point selection III additionally to all points lying on the farfield, the points which have a distance between 0 and $5.0 \cdot 10^{-5}$ grid units from the surface of the airfoil[1] are chosen. This region is part of the boundary layer. Thus, point selection III gives an insight into the influence the boundary layer has on the solution of the MPE. In this setting a total of 12,940 points are considered.

Since it is well known that the boundary layer accounts for a large amount of the points of the computational grid, a lot of points have to be selected also for MPE when considering the boundary layer. For this reason, in point selection IV we choose those points which are right outside of the boundary layer. Obviously, these points are still close to the surface, but not as numerous in this region. For this particular test case, along with all farfield points, the points which have a distance between 0.02 and 0.5 units from the surface of the airfoil are selected. In this way 6,252 points are used in the MPE.

Furthermore, we choose another set of points, which is a bit further away from the boundary, but still close to the surface. More precisely, we choose all points having a distance between 0.1 and 1.0 units from the surface.

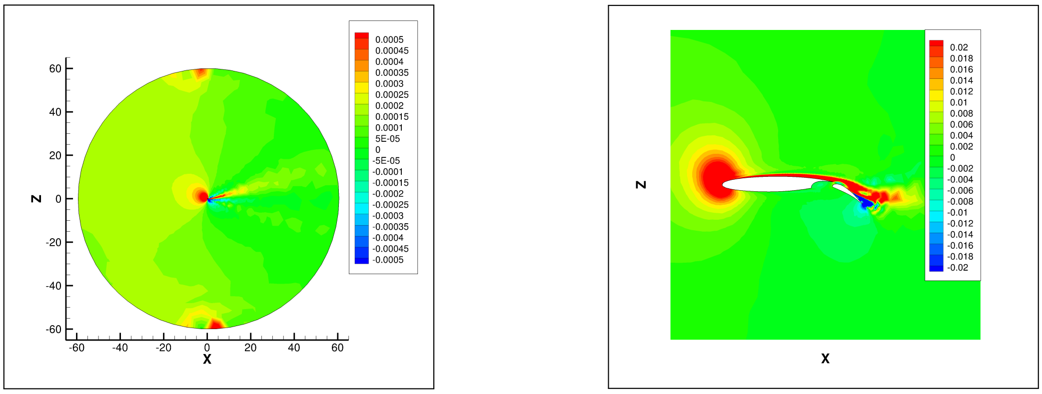
---

[1] Note that the airfoil has a reference length of 0.6 grid units.



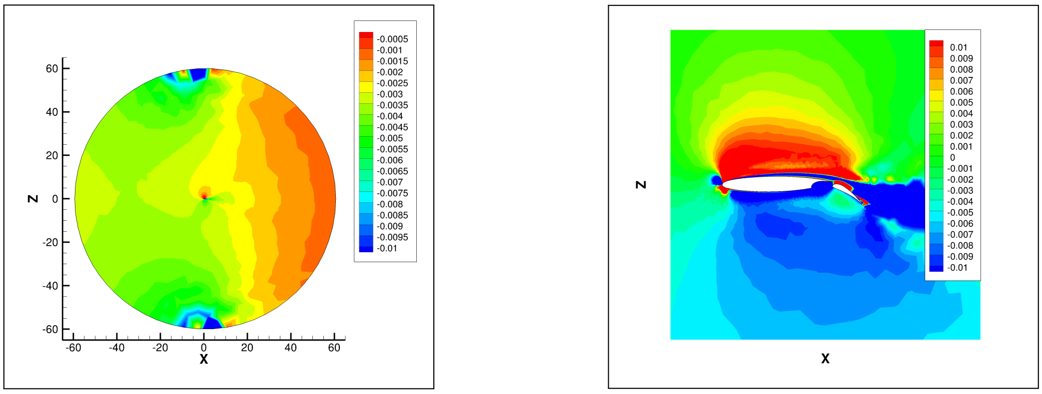
(a) Energy mode 1.

(b) Zoomed view of energy mode 1.



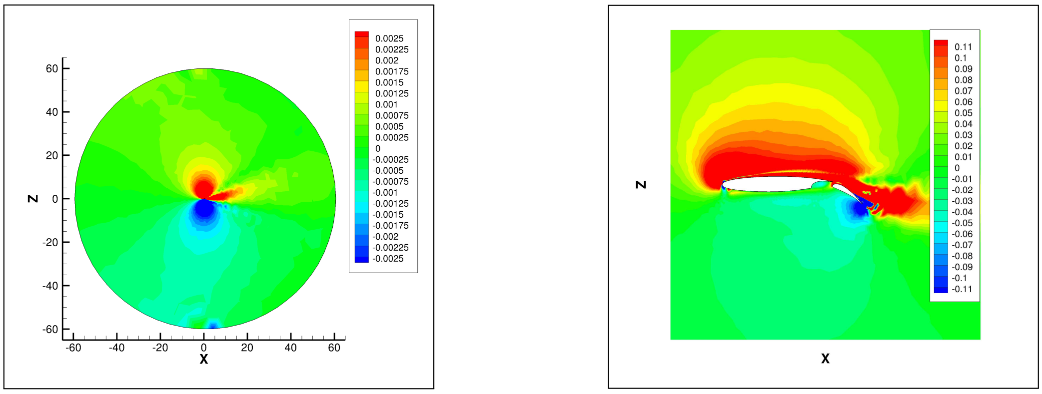
(c) Energy mode 2.

(d) Zoomed view of energy mode 2.



(e) Energy mode 3.

(f) Zoomed view of energy mode 3.



(g) Energy mode 4.

(h) Zoomed view of energy mode 4.

Figure 10.2: *The modes of the DLR-F15 take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*).*

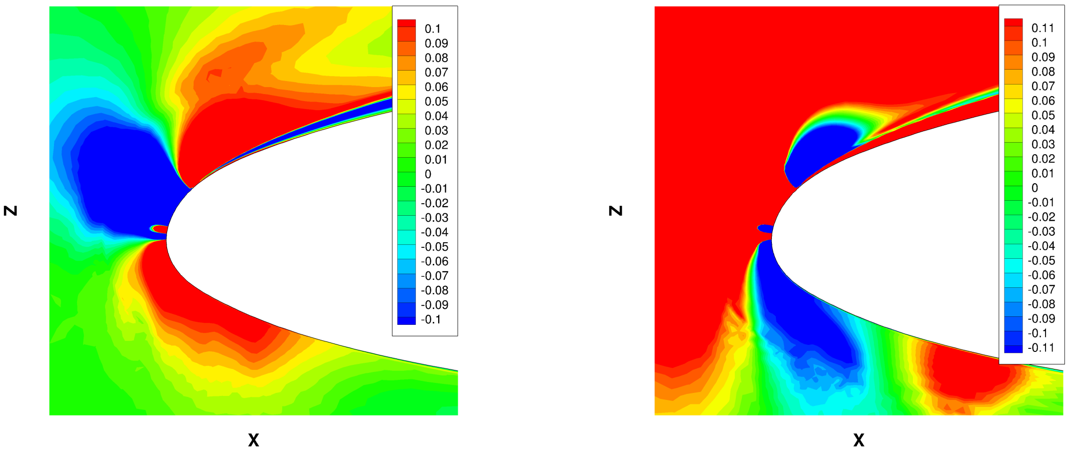
Finally, all points are chosen in selection VI. A summary of the different point selection settings is given in Table 10.4.

point selection	# of points	min distance	max distance
I (1 farfield point)	1	-	-
II (entire farfield)	60	99.0	101.0
III (FF + some BL points)	12,940	0.0	$5.0 \cdot 10^{-5}$
IV (FF + points close to BL)	6,252	0.02	0.5
V (FF + pts further from BL)	1,273	0.1	1.0
VI (all points)	50,809	0.0	101.0

Table 10.4: *The definition of the point selections for the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*), where min/max distance stands for the minimal and maximal distance from the surface of the configuration in grid units (length of the airfoil = 0.6 grid units). Note that FF and BL shall be abbreviations for the farfield and the boundary layer, respectively.*

Table 10.5 shows the lift and drag coefficients as well as the corresponding errors for the different point selections. It can be seen that the error is least for point selection scenarios IV and V, for which – additionally to the farfield – points outside of the boundary layer are chosen. In fact, scenario IV is best for the lift and V for the drag coefficient. The worst result is obtained when the boundary layer is included. This indicates that the turbulent region lacks spatial correlation in the POD basis with the rest of the flowfield.

In particular, note that the energy modes have a complex structure in the boundary layer (Figure 10.2). It can be seen in Figure 10.3 that the third and fourth energy mode have local minima and maxima in close proximity to one another on the leading edge of the airfoil.



(a) Energy mode 3. (b) Energy mode 4.

Figure 10.3: *The third and fourth energy mode on the leading edge of the take-off configuration.*

Note that it is not surprising that the missing point estimation model has problems capturing the small-scale phenomena in the boundary layer, since the low-dimensional basis of the reduced order model is not capable of accurately representing the small-scale turbulence. It is expected that only a very large number of modes would be able to capture the turbulence. This large number ($\gg 100$) of modes, however, would prohibit to obtain a solution faster than with a CFD solver.

Based on the number of residual evaluations, scenarios I and III are the most ineffective point selections. However, it has to be emphasized that a residual evaluation of scenario I is very efficient, since it has to be evaluated for a single point only.

point selection	residual eval's	c_l (error in %)	c_d (error in %)
I	33	2.696 (0.41%)	$4.864 \cdot 10^{-2}$ (0.37%)
II	13	2.696 (0.41%)	$4.865 \cdot 10^{-2}$ (0.39%)
III	22	2.708 (0.86%)	$4.967 \cdot 10^{-2}$ (2.5%)
IV	13	2.688 (0.11%)	$4.798 \cdot 10^{-2}$ (0.99%)
V	13	2.693 (0.30%)	$4.842 \cdot 10^{-2}$ (0.08%)
VI	13	2.690 (0.19%)	$4.816 \cdot 10^{-2}$ (0.62%)

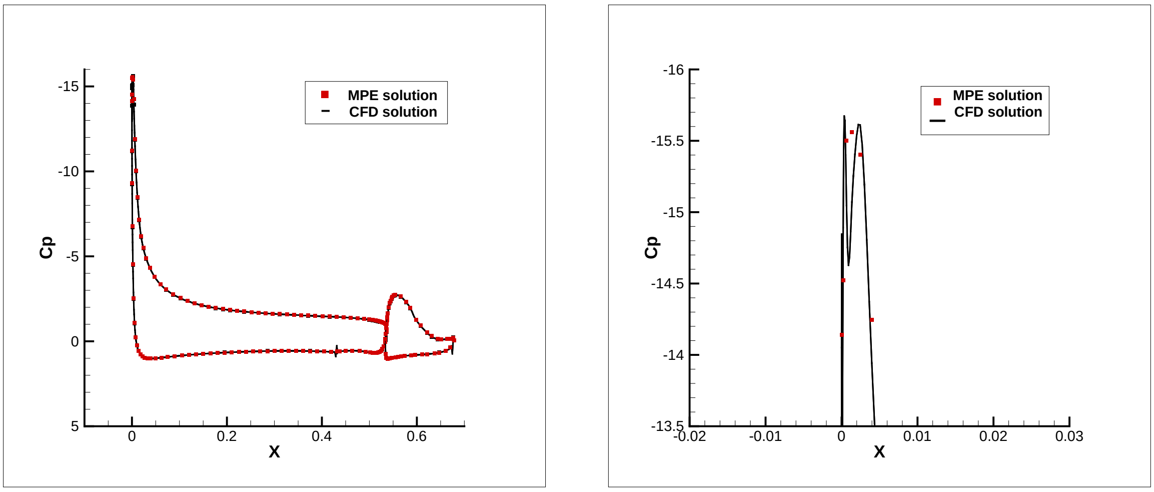
Table 10.5: *Lift and drag coefficients computed with MPE for different point selections for the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) at* $\alpha = 7°$. *Point selections I see Figure 10.1(b); selections II - V see Table 10.4.*

Note that when taking into account only the farfield points, a good result with an error less than one percent is achieved. Due to the fact that these points can also be easily identified, this is a practical approach in finding a good point selection strategy for this problem class.

We investigate the quality of the pressure distribution c_p on the surface of the MPE solution, where the residuals are evaluated at all farfield points. In Figure 10.4 both the reference c_p computed with CFD and that of the MPE using all four modes and with point selection II is shown. Obviously, they match very well and there is only a slight deviation near the stagnation point, where the pressure coefficient c_p is maximal (see Figure 10.4(b)).

In the end, note that if the snapshots are to be reconstructed with MPE, the results for the aerodynamic coefficients for the different point selections are in general better than for solutions at angles of attack, which are not included in the snapshot set. Numerical experiments have shown that there are only few exceptions to this. As the reconstruction of the snapshots is not of interest in practical applications, we will refrain from showing these results here.

Computational time: In the following we want to compare the computational time spend on the missing point estimation with that of the CFD computation. For the computations an AMD Athlon$^{\text{TM}}$ 64 X2 Dual Core Processor 4400+ with 2 GB RAM is used. The computer is operated under Debian GNU/LINUX.



(a) Full view.

(b) Zoomed view of the suction peak.

Figure 10.4: *The pressure distribution on the surface of the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*, and* $\alpha = 7°$*) for the MPE solution obtained using only farfield points in the residual evaluations.*

Table 10.6 shows both the time and the number of iterations[2] for the MPE and for CFD. Note that the CFD computation is initialized with the solution at an angle of attack of one degree below that of the computed solution, i. e. $\alpha = 6°$, and for which convergence is detected based on a Cauchy-type criterion for the aerodynamic coefficients as done for the snapshots.

model	iterations (residual eval's)	time (in CPU s)
MPE (farfield points)	13	< 9.864
CFD	1415	999.82

Table 10.6: *Computational time for the high-lift take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*, and* $\alpha = 7°$*) computed with the MPE using only the farfield points compared to CFD. The time for MPE includes the computation of POD, but not the generation of the snapshots. Since MPE is not efficiently implemented yet (see text below), the time is expected to be less. Hence the* $<$ *sign.*

It can be seen that both the number of iterations as well as the computational time spend on the CFD solution is about one hundred times larger than the equivalents of the missing point estimation. Thus, there is a large amount of time saved with the help of MPE.

It has to be noted that at this point the MPE is not efficiently implemented yet. In particular, in the current implementation, the residual is evaluated for every point and subsequently, filtered to the selected points. Therefore, the effect of reducing the number of equations to a small set of selected points is not yet incorporated in the above numbers. Hence, the computational time

---

[2] Note that number of iterations is not quite correct for the missing point estimation. Rather, the term number of residual evaluations describes the given number more accurately, since for the construction of the Jacobian in solving the reduced order model residual evaluations are needed, which are not iterations themselves. For CFD computations the number of iterations are in fact the number of pseudo-time steps.

is expected to be only a fraction of the given time. Note that 2.838 out of the total 9.864 CPU seconds are spent on reading the snapshots and computing the POD. Hence, an efficient implementation has substantial room for improvement.

Solutions at some other angles of attack: In order to prove that the model can be used for predicting different solutions, we compute the solutions at some intermediate angles of attack (AoA), that is at $\alpha \in \{1°, 3°, 5°\}$. For this, the same POD basis and in particular, the same number of modes $d = 4$ are used. In addition, the solution for an angle of attack outside of the range of the snapshots, i.e. $\alpha = 9°$, is considered. Note that this will be referred to as *extrapolation*, while for the previously mentioned angles the solutions are in the *interpolation* range. Table 10.7 shows the aerodynamic coefficients and their errors in comparison to the CFD solution.

AoA	c_l of MPE (error)	c_l of CFD	c_d of MPE (error)	c_d of CFD
1°	2.112 (0.61%)	2.125	$3.465 \cdot 10^{-2}$ (0.84%)	$3.436 \cdot 10^{-2}$
3°	2.310 (0.26%)	2.304	$3.828 \cdot 10^{-2}$ (0.36%)	$3.814 \cdot 10^{-2}$
5°	2.495 (0.12%)	2.492	$4.234 \cdot 10^{-2}$ (0.37%)	$4.250 \cdot 10^{-2}$
7°	2.696 (0.41%)	2.685	$4.865 \cdot 10^{-2}$ (0.39%)	$4.846 \cdot 10^{-2}$
9°	2.920 (1.96%)	2.864	$5.932 \cdot 10^{-2}$ (16.06%)	$7.067 \cdot 10^{-2}$

Table 10.7: *Lift and drag coefficients of the solutions at different angles of attack (AoA) for the high-lift take-off configuration (Re = 10.5 million, $M_\infty = 0.22$) computed with MPE using all modes ($d = 4$) and farfield points compared to CFD.*

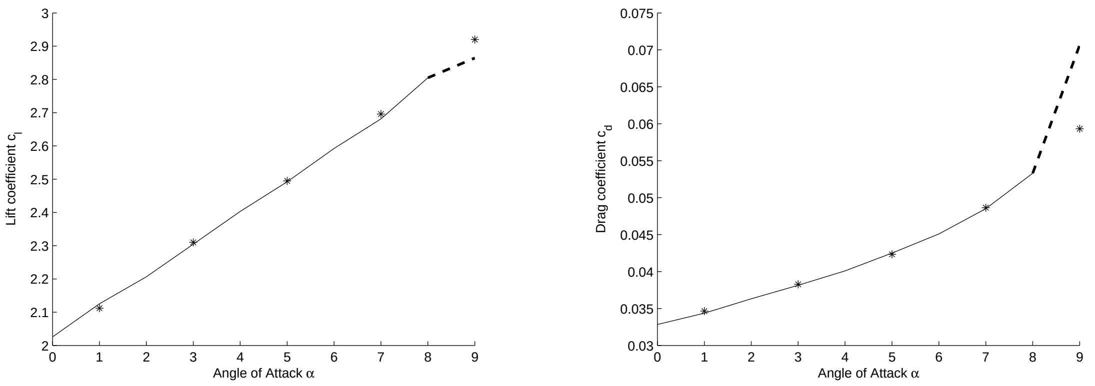
Note that all relative errors are less than one percent in the interpolation range, while for the extrapolation of $\alpha = 9°$ the error is much higher. As a matter of fact, the error for the drag coefficient is with 16 percent unacceptably high. The most important reason for this is that the static stall region is reached, for which the input-output behavior of the system becomes highly nonlinear and this behavior is not captured in the snapshots. This can be seen in Figure 10.5, in which plots of the lift and drag coefficient over the angle of attack are shown. Note that the MPE solutions given by the asterisks closely match the CFD reference solution in the interpolation range.

Comparison with Thin Plate Spline (TPS) interpolation: In the following the results of MPE shall be compared to interpolation. The interpolation is carried out as follows: At first, the POD coefficients of the snapshots $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m$ are computed via

$$a_i(\alpha_j) = (\boldsymbol{u}_i, \boldsymbol{w}_j)_{L_2}, \quad i = 1, \ldots, d, \quad j = 1, \ldots, m \tag{10.2}$$

Then these coefficients are interpolated to obtain suitable coefficients $a_i(\alpha^*)$ for the desired angle of attack α^*. In this thesis Thin Plate Spline (TPS) interpolation is used [19].

In Table 10.8 the aerodynamic coefficients as well as their relative errors are given for TPS interpolation. It can be observed that the coefficients are well approximated for the two angles of



(a) The Lift coefficient.

(b) The Drag coefficient.

Figure 10.5: *The aerodynamic coefficients plotted over the angle of attack for the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*). The line represents the coefficients of the CFD reference solution and the asterisks represent those of the solution of MPE. Note that the solid line represents the interpolation range, while the dashed lined represents the extrapolation range.*

attack $\alpha = 3°$ and $\alpha = 5°$. In fact, the interpolation yields better results than the MPE solution. However, for the other two angles, $\alpha = 1°$ and $\alpha = 7°$, the MPE results are significantly better. This is due to the fact that these angles are close to the boundary of the interpolation range, i.e. there is only one neighboring snapshot with a lower or higher angle of attack.

This points out an important property of the MPE reduced order system: It approximates the solution in the interpolation range better in a global sense. That is, the maximal error of MPE at the considered angles with less than one percent is smaller than TPS interpolation with a maximal error of about three percent. This means that MPE can be trusted throughout the interpolation range, while TPS interpolation yields in some regions better results. Note that TPS interpolation also yields better results for the extrapolation.

AoA	c_l of TPS (error)	c_l of CFD	c_d of TPS (error)	c_d of CFD
1°	2.115 (0.47%)	2.125	$3.334 \cdot 10^{-2}$ (2.97%)	$3.436 \cdot 10^{-2}$
3°	2.304 (0.00%)	2.304	$3.807 \cdot 10^{-2}$ (0.18%)	$3.814 \cdot 10^{-2}$
5°	2.497 (0.20%)	2.492	$4.245 \cdot 10^{-2}$ (0.12%)	$4.250 \cdot 10^{-2}$
7°	2.697 (0.45%)	2.685	$4.754 \cdot 10^{-2}$ (1.90%)	$4.846 \cdot 10^{-2}$
9°	2.910 (1.61%)	2.864	$6.555 \cdot 10^{-2}$ (7.24%)	$7.067 \cdot 10^{-2}$

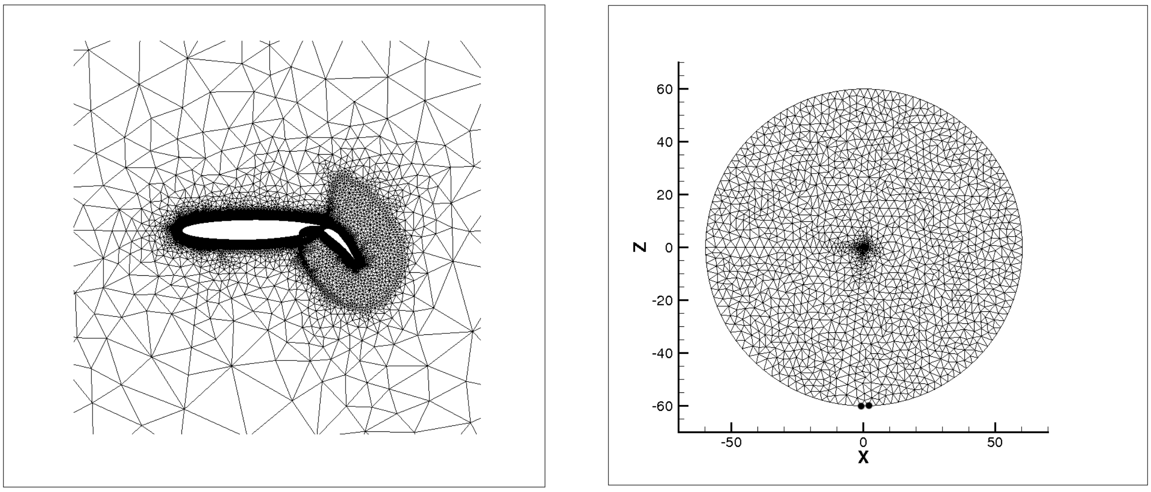
Table 10.8: *Lift and drag coefficients of the solutions at different angles of attack (AoA) for the high-lift take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) obtained with TPS interpolation and compared with CFD reference solution.*

10.2 Two-dimensional DLR-F15 airfoil in landing conditions

As a second test case we will consider the DLR-F15 high-lift configuration under landing conditions. The viscous flow around a section cut of this configuration modeled by the Navier-Stokes equations is considered at an atmospheric pressure of 101325 Pa, a temperature of 15°C ($288.15\,K$) and a speed of $|\boldsymbol{v}_\infty| = 53$ m/s, which corresponds to a Mach number of $M_\infty = 0.1558$. This setting correlates to a Reynolds number of 2.177 million.

Snapshots are computed for angles of attack of $\alpha \in \{0°, 2°, 4°, 6°, 8°\}$. Once again, the computations are carried out with the TAU code [22, 21], this time using a fixed number of iterations of 10000 for $\alpha = 0°$ and $\alpha = 2°$. In this way the norm of the residuals is dropped by at least seven orders of magnitude. The other snapshots are obtained by initializing the computation with the solution from one angle of attack below the one, which is to be computed, and using the convergence criterion based on the aerodynamic coefficients with the same settings as described in Subsection 10.1.

Again, the grid is a hybrid one, structured in the region of the boundary layer and unstructured elsewhere. It is shown in Figure 10.6 and contains $N = 73,474$ mesh points.



(a) Detailed view of the computational grid close to the surface

(b) Full view with marked points for point selection setting I of Table 10.12.

Figure 10.6: *The hybrid grid of the high-lift landing configuration consisting of* $N = 73,474$ *points.*

Influence of the number of used modes: After computing the POD, we take a look at the relative energy of the modes as defined in equation (10.1). It is presented in Table 10.9. Once again, the first mode contains 99.9% of the energy.

Next, we want to investigate the impact of the truncation level for this test case. Table 10.10 displays the lift and drag coefficient as well as the corresponding errors. It can be seen that the more modes are considered, the better is the prediction of the lift coefficients. For the drag coefficient the error is less for two than for three modes, but it is least when all modes are considered. For this reason we will carry out all computations with all modes henceforth.

j	1	2	3	4	5
$E(j)$	$9.99579 \cdot 10^{-1}$	$4.209600 \cdot 10^{-4}$	$1.88304 \cdot 10^{-7}$	$9.33441 \cdot 10^{-8}$	0

Table 10.9: *Relative energy content of the modes corresponding to the snapshots at* $\alpha \in \{0°, 2°, 4°, 6°, 8°\}$ *of the high-lift landing configuration (*$Re = 2.177$ *million,* $M_\infty = 0.1558$*).*

# of modes	residual eval's	c_l (error in %)	c_d (error in %)
$d = 1$	7	2.539 (0.27%)	$8.307 \cdot 10^{-2}$ (12.61%)
$d = 2$	8	2.541 (0.20%)	$9.513 \cdot 10^{-2}$ (0.07%)
$d = 3$	10	2.542 (0.16%)	$9.519 \cdot 10^{-2}$ (0.14%)
$d = 4$	11	2.544 (0.08%)	$9.510 \cdot 10^{-2}$ (0.04%)
CFD	1346	2.546	$9.506 \cdot 10^{-2}$

Table 10.10: *Lift and drag coefficients of the landing configuration (*$Re = 2.177$ *million,* $M_\infty = 0.1558$*) computed with MPE, where the residual evaluations are restricted to the farfield points. The aerodynamic coefficients are predicted at* $\alpha = 3°$ *for different numbers of modes* d *and compared to the CFD reference solution at the same angle.*

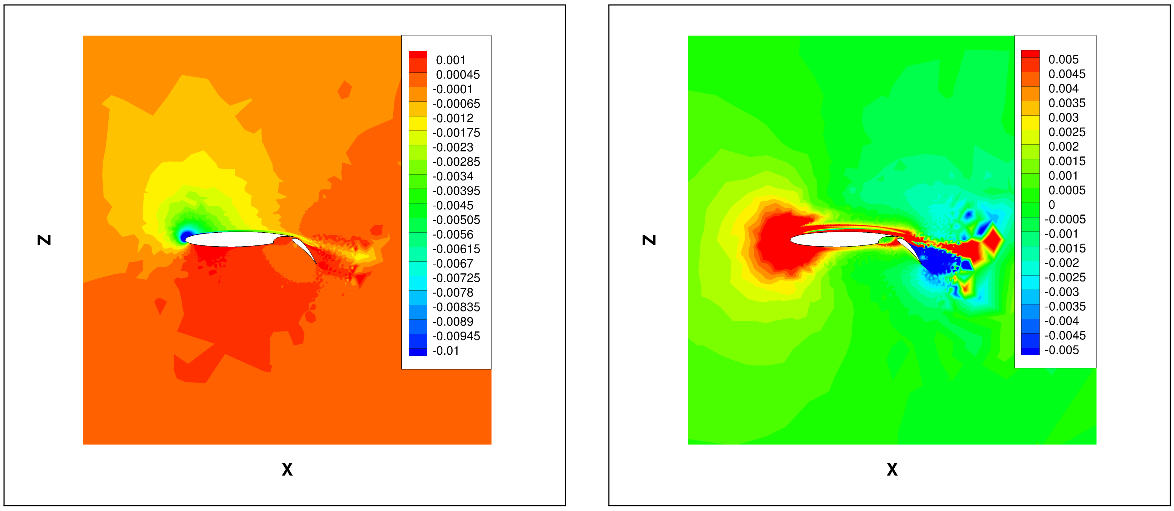
As it has been done for the previous test case, the POD modes shall be investigated. The corresponding coefficients are given in Table 10.11. It can be seen that the coefficients are similar for the first two modes, irrespective of the number of used modes.

Figure 10.7 shows the energy modes of this test case.

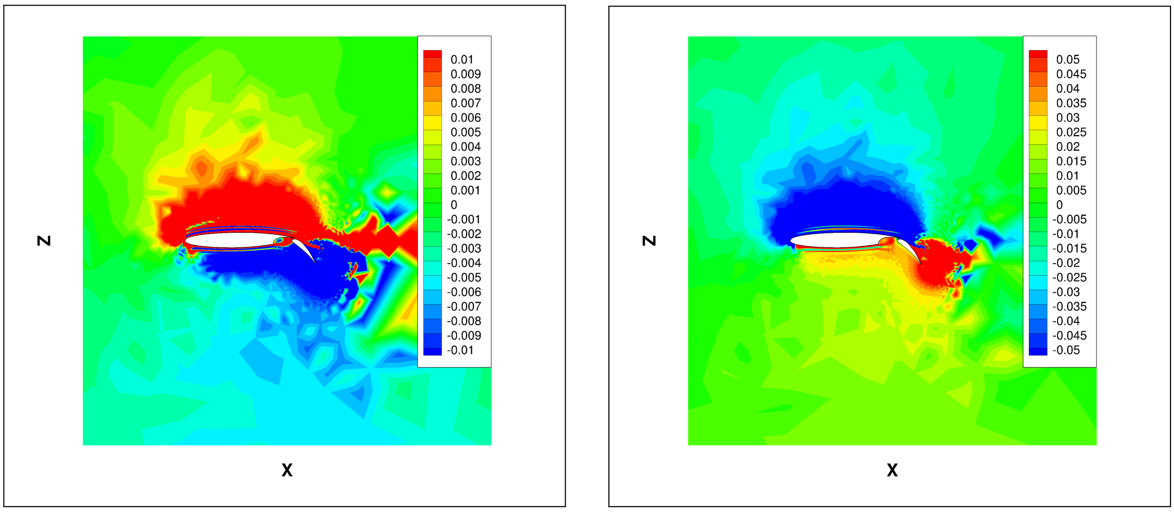
Influence of the selection matrix: For investigating the influence of the point selection on the solution of the MPE, we use similar point selection settings as in the previous test case. However, they have to be adjusted to the computational grid since, for example, the boundary layer has a different thickness for this mesh. Therefore, the minimal and maximal distance for the definition of the points for setting IV are different. Specifics are given in Table 10.12. Note that point selection setting I is given by the points marked in Figure 10.6(b). Once again, the points are chosen such that the onflow is parallel to the tangent of the computational domain.

# of modes	1	2	3	4
$d = 1$	-0.73204			
$d = 2$	-0.73223	0.044396		
$d = 3$	-0.73223	0.044520	-0.00077921	
$d = 4$	-0.73228	0.044442	-0.00044296	0.00034954

Table 10.11: *The POD coefficients for the modes obtained with MPE, where the residual evaluations are restricted to the farfield points, using different numbers of modes* d*.*



(a) Energy mode 1. (b) Energy mode 2.



(c) Energy mode 3. (d) Energy mode 4.

Figure 10.7: *The energy modes of the landing configuration (Re* $= 2.177$ *million,* $M_\infty = 0.1558$*).*

point selection	# of points	min distance	max distance
I (2 farfield points)	2	-	-
II (entire farfield)	125	99.0	101.0
III (FF + some BL points)	10,974	0.0	$5.0 \cdot 10^{-5}$
IV (FF + points close to BL)	2,360	0.07	0.5
V (FF + points further from BL)	1,882	0.1	1.0
VI (all points)	73,474	0.0	101.0

Table 10.12: *The definition of the point selections for the landing configuration (Re* = 2.177 *million,* $M_\infty = 0.1558$*), where min/max distance stands for the minimal and maximal distance from the surface of the configuration in spatial units (length of the airfoil = 0.6 units). Note that FF and BL shall be abbreviations for the farfield and the boundary layer, respectively.*

Table 10.13 presents the aerodynamic coefficients for the different point selection settings. As in the previous test case setting I, which is given by two of the farfield points shown in Figure 10.6(b), gives good results, but needs the most residual evaluations. Since the cost for one residual evaluation is least for this setting, it is expected that this will compensate for the high number of evaluations.

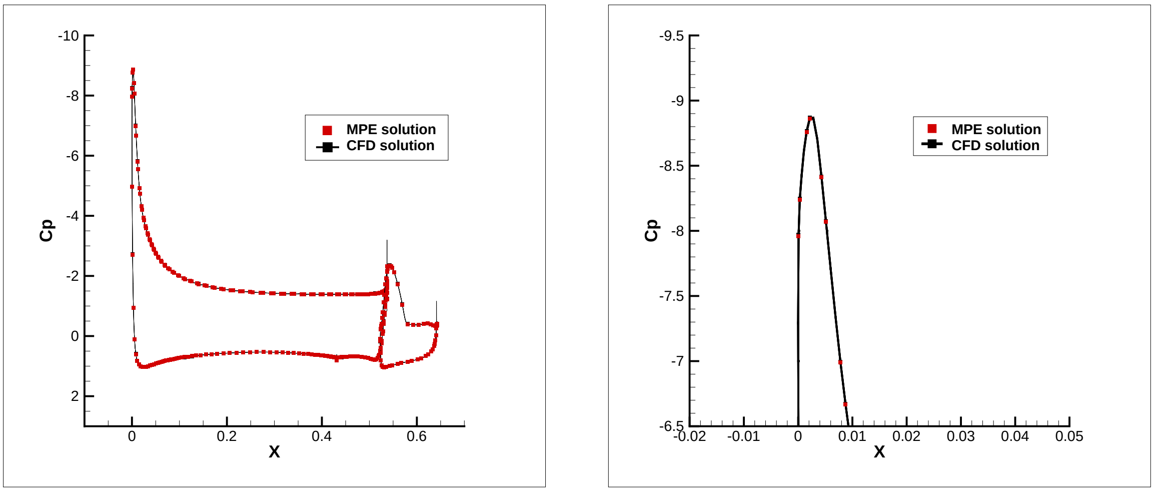
point selection	residual eval's	c_l (error in %)	c_d (error in %)
I	17	2.541 (0.20%)	$9.533 \cdot 10^{-2}$ (0.28%)
II	11	2.544 (0.08%)	$9.510 \cdot 10^{-2}$ (0.04%)
III	11	2.544 (0.08%)	$9.511 \cdot 10^{-2}$ (0.05%)
IV	12	2.544 (0.08%)	$9.510 \cdot 10^{-2}$ (0.04%)
V	12	2.544 (0.08%)	$9.510 \cdot 10^{-2}$ (0.04%)
VI	13	2.545 (0.04%)	$9.511 \cdot 10^{-2}$ (0.05%)
CFD	1346	2.546	$9.506 \cdot 10^{-2}$

Table 10.13: *Lift and drag coefficients computed with MPE for different point selections for the landing configuration (Re* = 2.177 *million,* $M_\infty = 0.1558$*, and* $\alpha = 3°$*). Point selections I see Figure 10.6(b); II - V see Table 10.12*

For all other point selections, both the lift and drag coefficient are mostly the same. As before, since the farfield points can easily be identified, this is the most effective strategy.

In Figure 10.8 the pressure distribution on the surface of the MPE solution – considering only the farfield points – and of the CFD reference solution is visualized. Obviously, both match very closely.

Computational time: Next, we want to focus on the computational time saved when employing the MPE. For the computations an AMD Athlon$^{\text{TM}}$ 64 X2 Dual Core Processor 4400+ with 2 GB RAM is used. The computer is operated under Debian GNU/LINUX.



(a) Full view.

(b) Zoomed view of the suction peak.

Figure 10.8: *The pressure distribution on the surface of the landing configuration (Re* $= 2.177$ *million,* $M_\infty = 0.1558$*, and* $\alpha = 3°$*) for the MPE solution obtained using only farfield points in the residual evaluations.*

As before, the CFD reference solution is initialized with the solution at an angle of attack one degree below the one to be computed. That is, it is started from the solution at $\alpha = 2°$. Again, the solution is considered converged, if the aerodynamic coefficients are not significantly changing over an interval of one hundred iterations.

In Table 10.14 both the number of iterations as well as the computational time for the MPE and the CFD computation is given. It can be observed that the CFD computation is more than a hundred times slower and needs more than a hundred times more iterations. In fact, the MPE model needs 29.936 CPU seconds to converge, including 5.058 CPU seconds for reading the snapshots and computing the POD. This is the average obtained from 10 runs. Once again, since most time of the MPE is spent on the simulation itself, an efficient implementation, which evaluates the residual sparsely, is expected to improve upon on the overall time.

model	iterations (residual eval's)	time (in CPU s)
MPE (farfield points)	11	< 23.343
CFD	1346	3582.6

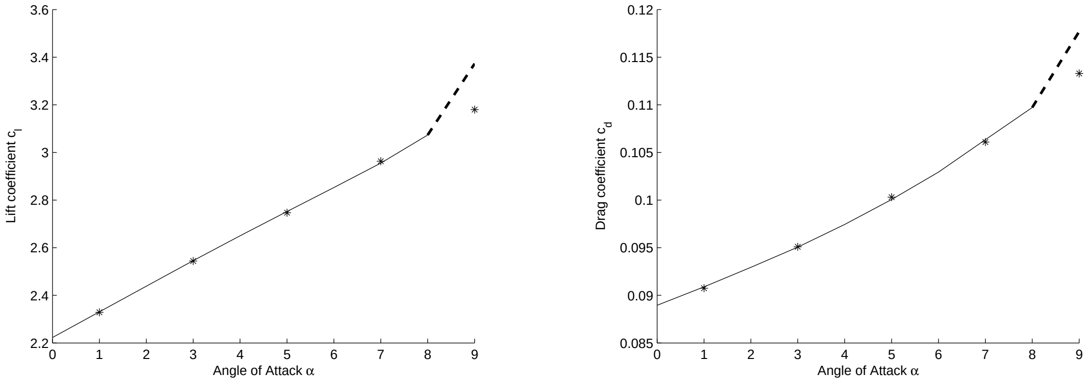
Table 10.14: *Computational time for the landing configuration (Re* $= 2.177$ *million,* $M_\infty = 0.1558$*, and* $\alpha = 3°$*) computed with the MPE using only the farfield points and with CFD. The time for MPE includes the computation of POD, but not the generation of the snapshots. Since MPE is not efficiently implemented yet, the time is expected to be less. Hence the* $<$ *sign.*

Solutions at other angles of attack: In the following we want to take a look at the solution of some intermediate angles of attack, namely $\alpha \in \{1°, 5°, 7°\}$. Also, the solution at $\alpha = 9°$ is considered, which lies outside of the interpolation range. Table 10.15 gives an overview of the aerodynamic coefficients and the errors. In fact, it can be observed that all relative errors are below 0.27% in the interpolation range.

AoA	c_l of MPE (error)	c_l of CFD	c_d of MPE (error)	c_d of CFD
1°	2.329 (0.09%)	2.331	$9.075 \cdot 10^{-2}$ (0.15%)	$9.089 \cdot 10^{-2}$
3°	2.544 (0.08%)	2.546	$9.510 \cdot 10^{-2}$ (0.04%)	$9.506 \cdot 10^{-2}$
5°	2.747 (0.18%)	2.752	$1.003 \cdot 10^{-1}$ (0.19%)	$1.001 \cdot 10^{-1}$
7°	2.963 (0.27%)	2.955	$1.061 \cdot 10^{-1}$ (0.19%)	$1.063 \cdot 10^{-1}$
9°	3.180 (5.72%)	3.373	$1.133 \cdot 10^{-1}$ (3.74%)	$1.177 \cdot 10^{-1}$

Table 10.15: *Lift and drag coefficients of the solutions for different angles of attack of the landing configuration (Re* $= 2.177$ *million,* $M_\infty = 0.1558$*) computed with MPE using all farfield points and with CFD.*

For the extrapolation the errors are much larger. Again, the extrapolated region is close to the static stall region and, therefore, the input-output behavior becomes strongly nonlinear. This can be seen in Figure 10.9, which shows the lift and drag versus the angle of attack. Since the nonlinear behavior is not captured in the snapshots, the error is quite large. In the interpolation range, on the other hand, the lift and drag coefficient of the MPE solution (represented by the asterisks) match those of the reference solution (given by the line).



(a) The lift coefficient.

(b) The drag coefficient.

Figure 10.9: *The aerodynamic coefficients plotted over the angle of attack for the landing configuration (Re* $= 2.177$ *million,* $M_\infty = 0.1558$*). The line represents the coefficients of the CFD solution and the asterisks represent those of the solution of MPE.*

Comparison with Thin Plate Spline (TPS) interpolation: In the end, the MPE results shall be compared to TPS interpolation. Again, it can be seen that interpolation performs worst, if it is carried out for a parameter close to the boundary of the interpolation range. MPE yields significantly better results for these angles of attack ($\alpha = 1°$ and $\alpha = 7°$). This supports once more the fact that the MPE system has a better global approximation quality than interpolation. Note that the error in extrapolated solution at $\alpha = 9°$ is similar for MPE and TPS.

AoA	c_l of MPE (error)	c_l of CFD	c_d of MPE (error)	c_d of CFD
1°	2.330 (0.04%)	2.331	$8.941 \cdot 10^{-2}$ (1.63%)	$9.089 \cdot 10^{-2}$
3°	2.545 (0.04%)	2.546	$9.511 \cdot 10^{-2}$ (0.05%)	$9.506 \cdot 10^{-2}$
5°	2.751 (0.04%)	2.752	$1.002 \cdot 10^{-1}$ (0.10%)	$1.001 \cdot 10^{-1}$
7°	2.962 (0.24%)	2.955	$1.047 \cdot 10^{-1}$ (1.51%)	$1.063 \cdot 10^{-1}$
9°	3.184 (5.60%)	3.373	$1.223 \cdot 10^{-1}$ (3.91%)	$1.177 \cdot 10^{-1}$

Table 10.16: *Lift and drag coefficients of the solutions at different angles of attack (AoA) for the landing configuration (*$Re = 2.177$ *million,* $M_\infty = 0.1558$*) obtained with TPS interpolation and compared with the CFD reference solution.*

10.3 Three-dimensional DLR-F12 configuration

In this section it shall be demonstrated that the proposed method also works for complex three-dimensional aircraft configurations. As a test case the DLR-F12 is used. It is a three-dimensional wing-body airplane configuration with a vertical and horizontal stabilizer (see Figure 10.10). Some of the results that will be shown here, have been published in [65].

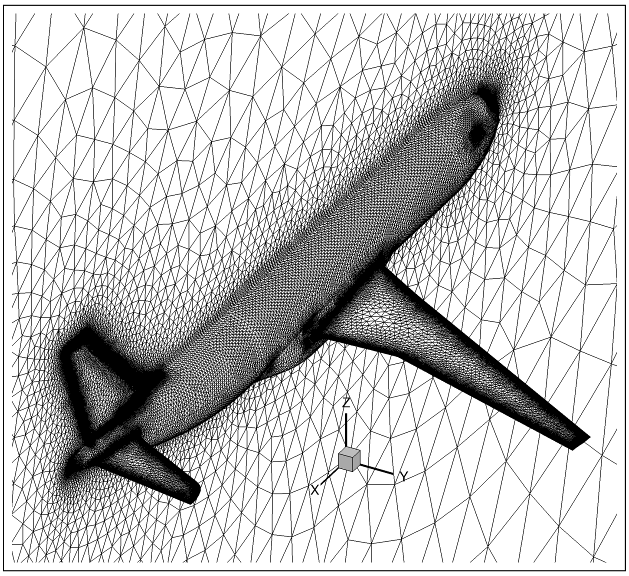


Figure 10.10: *The computational grid of the DLR-F12 configuration.*

The computational domain around this configuration is discretized with an unstructured grid consisting of about 670 thousand grid points ($N = 669,062$). A fragment of this computational grid is illustrated in Figure 10.10.

In this test case the inviscid flow past the DLR-F12 configuration at a freestream Mach number of $M_\infty = 0.2$ is modeled. Snapshots for a range of different angles of attack at this subsonic flow velocity, namely $\alpha \in \{0°, 2°, 4°, 6°, 8°\}$, are computed to simulate the flow at an intermediate angle of attack of $\alpha = 5°$. The computations are carried out until ten thousand iterations are reached, in which case the residuals are smaller than 10^{-14}.

Note that the flow is computed with the Euler equations to keep the number of grid points small enough such that the computation can be carried out on a desktop computer.

Influence of the number of used modes: At first, investigate the number of modes used for the reduced order model. To this end, consider the relative energy content. Table 10.17 suggests that, as in the previous examples, the first mode carries more than 99.9% of the energy.

j	1	2	3	4	5
$E(j)$	$9.99560 \cdot 10^{-1}$	$4.40291 \cdot 10^{-4}$	$1.81356 \cdot 10^{-7}$	$1.04662 \cdot 10^{-8}$	0

Table 10.17: *Relative energy content of the modes corresponding to the snapshots at* $\alpha \in \{0°, 2°, 4°, 6°, 8°\}$ *of the DLR-F12 configuration at* $M_\infty = 0.2$.

In Table 10.18 the aerodynamic coefficients for the different choices of the number of modes are given. In each of these simulations the points, at which the residual is evaluated, are the same. In fact, only the farfield points are used. It can be seen that for accurate predictions of the drag coefficient at least two modes have to be chosen. With two modes, the drag coefficient is slightly overestimated by the MPE, while for three or four modes it is underestimated.

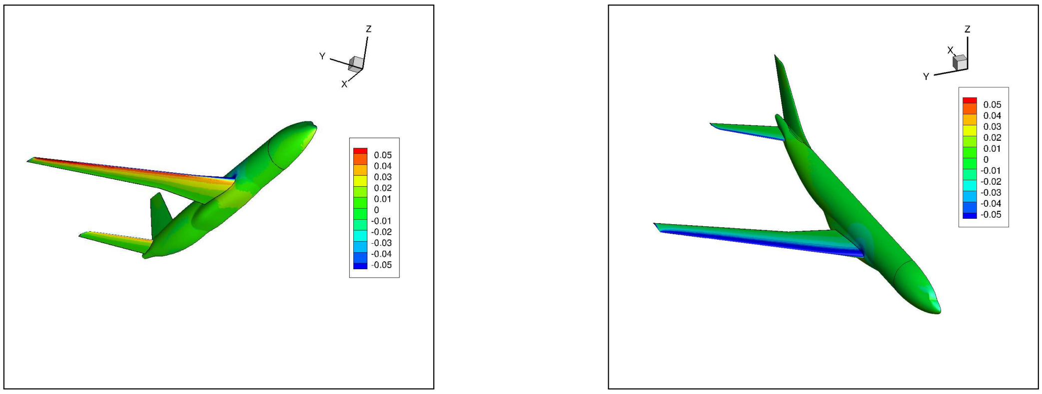
# of modes	residual evaluations	c_l (error in %)	c_d (error in %)
$d = 1$	8	$5.803 \cdot 10^{-1}$ (0.41%)	$1.449 \cdot 10^{-2}$ (26.82%)
$d = 2$	10	$5.827 \cdot 10^{-1}$ (0.0%)	$1.983 \cdot 10^{-2}$ (0.15%)
$d = 3$	14	$5.827 \cdot 10^{-1}$ (0.0%)	$1.975 \cdot 10^{-2}$ (0.25%)
$d = 4$	17	$5.827 \cdot 10^{-1}$ (0.0%)	$1.975 \cdot 10^{-2}$ (0.25%)
CFD	2000	$5.827 \cdot 10^{-1}$	$1.980 \cdot 10^{-2}$

Table 10.18: *Lift and drag coefficients of the DLR-F12 configuration at* $M_\infty = 0.2$ *computed with MPE, where the residual evaluations are restricted to the farfield points. The aerodynamic coefficients are predicted at* $\alpha = 5°$ *for different numbers of modes* m *and compared to the CFD reference solution at the same angle.*

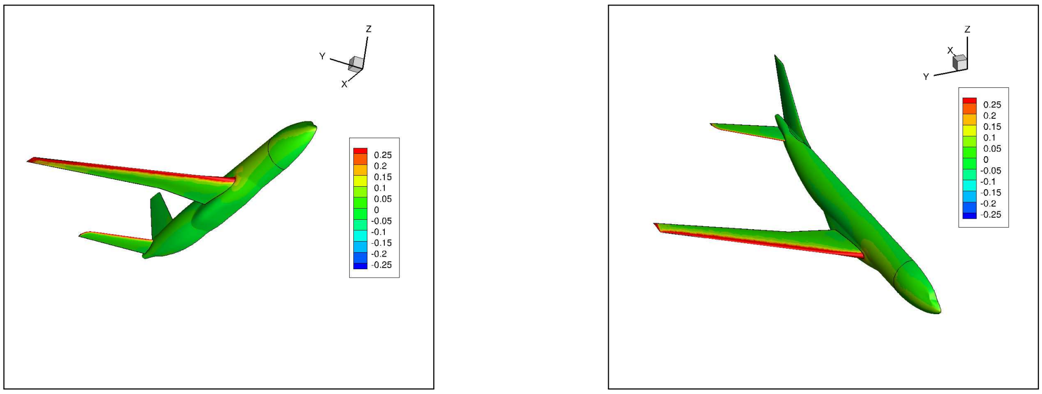
In order to find an explanation for this, we will look at the modes of the configuration. They are depicted in Figure 10.11. The corresponding POD coefficients are given in Table 10.19.

Note that the coefficients for the first two modes differ only slightly, irrespective of how many modes are used. This is not the case for those of the last two modes.

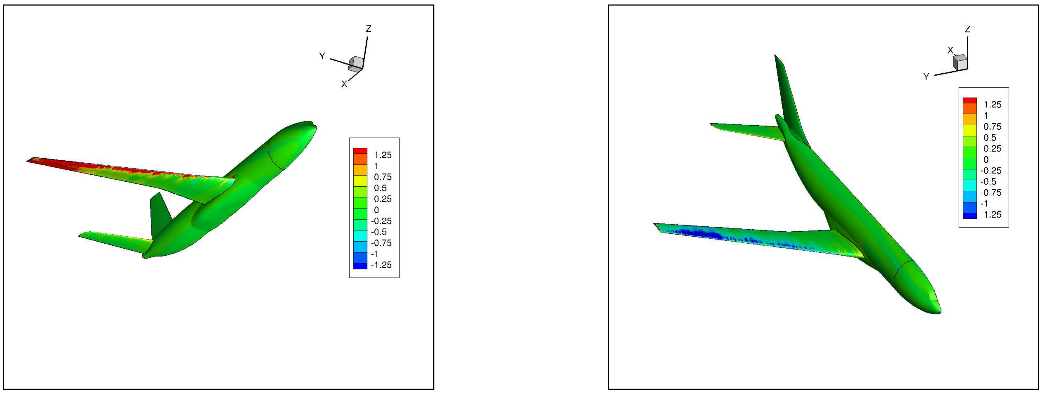
When looking at the first mode it can be observed that the values of the energy modes on the surface of the configuration are high on the lower side and low on the upper side of the wing. Thus, if this mode is multiplied with a positive coefficient as it is done for this computation (see



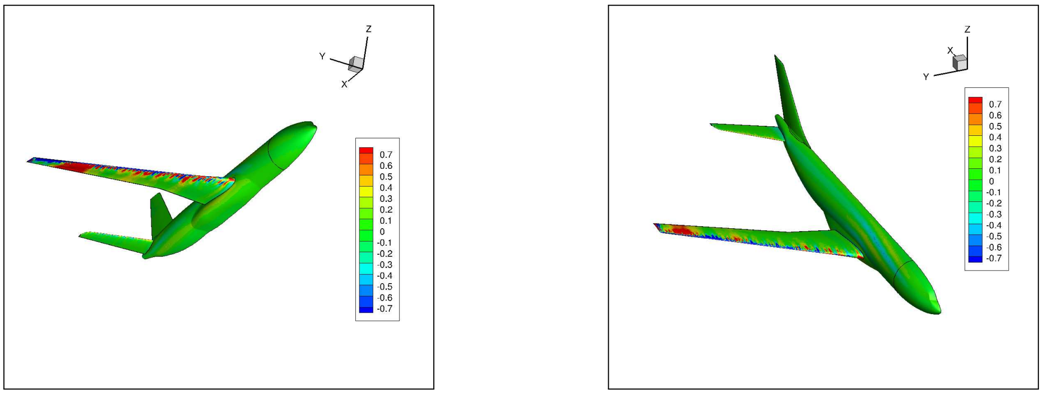
(a) Energy mode 1. (b) Energy mode 1.



(c) Energy mode 2. (d) Energy mode 2.



(e) Energy mode 3. (f) Energy mode 3.



(g) Energy mode 4. (h) Energy mode 4.

Figure 10.11: *The energy modes on the surface of the DLR-F12 configuration at* $M_\infty = 0.2$.

# of modes	1	2	3	4
$d = 1$	$8.70983 \cdot 10^{-2}$			
$d = 2$	$8.70625 \cdot 10^{-2}$	$5.40387 \cdot 10^{-3}$		
$d = 3$	$8.70412 \cdot 10^{-2}$	$5.40947 \cdot 10^{-3}$	$-1.9876 \cdot 10^{-4}$	
$d = 4$	$8.70519 \cdot 10^{-2}$	$5.40794 \cdot 10^{-3}$	$-1.6064 \cdot 10^{-4}$	$3.22069 \cdot 10^{-5}$

Table 10.19: *The POD coefficients for the modes obtained with MPE, where the residual evaluations are restricted to the farfield points, using different numbers of modes d.*

Table 10.19), the lift is increased.[3] Consequently, the first mode is very suitable for adjusting the lift. This explains the good approximation of the lift and bad approximation of the drag when using only one mode.

The second mode, on the other hand, is characterized by high numerical values on the leading edge of the wing and tailplane. As a result, the second mode is well suited for adjusting the drag, which explains the good approximation of the drag coefficient, if two modes are used.

While the first two modes can be characterized by a gradual distribution, this is not the case for the third and fourth mode. In particular, the last mode features local minima and maxima on the wing, which are in close proximity to each other. The rather unsmooth behavior of the last two modes provide a possible explanation why the use of more than two modes actually yields worse results.

Nevertheless, the fact that using two modes is better than using more modes is unexpected and undesirable. In fact, by setting the last two coefficients to zero, the method could achieve the same results as with two modes. However, it yields coefficients, which result in greater errors in the aerodynamic coefficients. Therefore, the finding of the root of the projected residual is not a method that will give an optimal approximation of the aerodynamic coefficients. In Chapter 11, this will be analyzed in detail.

Influence of the selection matrix: Next, the point selection for this three-dimensional test case shall be analyzed. Unlike in the other test cases, both two and four modes are considered. This is because using two modes yields the best approximation of the drag coefficient in the tests of Table 10.18.

Since in this test case an inviscid flow is computed, there is no boundary layer. Altogether four different point selections are considered: The first uses only the points on the farfield boundary. The second one uses additionally to the farfield those points lying in close proximity to the surface. In this way, a great majority of the points are selected. The third point selection uses additionally to the farfield some points, which are further away from the surface. Finally, the fourth consists of all points. Specifics on the definition of the point selections can be found in Table 10.20. In particular, the distance from the surface is given as well as the number of

---

[3] This is due to the fact that since the energy is coupled to the pressure through the equation of state (3.17), an increase in the energy values results in higher pressure values. These in turn influence the lift.

points for each selection.

point selection	# of points	min distance	max distance
I (entire farfield)	336	-	-
II (FF + points close to surface)	633,281	0.0	0.05
III (FF + points further from surface)	16,279	0.05	0.1
IV (all points)	669,062	0.0	100.0

Table 10.20: *The definition of the point selections for the DLR-F12 configuration at* $M_\infty = 0.2$*, where min/max distance stands for the minimal and maximal distance from the surface of the configuration in spatial units (length of the airfoil = 1.0 units). Note that FF shall be an abbreviation for the farfield.*

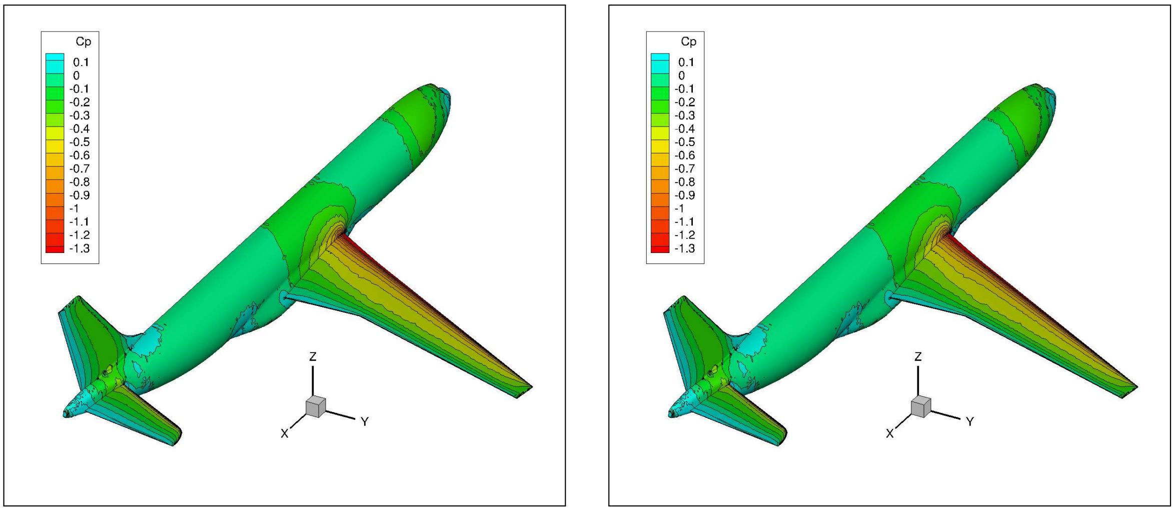
In Table 10.21 the lift and drag coefficient for each of the point selections, using two as well as four modes, is presented. If only two modes are employed, all point selections yield the same aerodynamic coefficients except when all points are selected, in which case, the drag coefficient is approximated slightly worse. However, the error in the drag coefficient is increased by less than one per mille and therefore it is insignificant.

In case four modes are used both with selection I and III, good approximations of the lift and drag coefficient are obtained. For point selection II and IV there is no convergence. As a matter of fact, after the first iteration of the Dog Leg method, an unphysical flow is obtained. Therefore, the residual could not be computed. This suggests that the points very close to the surface lack spatial correlation with the rest of the flow field. In fact, this is reflected in the afore-mentioned unsmooth behavior on the surface of the airfoil in the higher modes, especially the last one.

point selection	# modes	residual eval's	c_l (error in %)	c_d (error in %)
I, II, III	d=2	10	$5.827 \cdot 10^{-1}$ (0.0%)	$1.983 \cdot 10^{-2}$ (0.15%)
IV	d=2	10	$5.827 \cdot 10^{-1}$ (0.0%)	$1.984 \cdot 10^{-2}$ (0.20%)
I	d=4	17	$5.827 \cdot 10^{-1}$ (0.0%)	$1.975 \cdot 10^{-2}$ (0.25%)
II, IV	d=4	-	-	-
III	d=4	17	$5.827 \cdot 10^{-1}$ (0.0%)	$1.976 \cdot 10^{-2}$ (0.20%)
CFD	-	2000	$5.827 \cdot 10^{-1}$	$1.980 \cdot 10^{-2}$

Table 10.21: *Lift and drag coefficients computed with MPE for different point selections for the DLR-F12 configuration at* $M_\infty = 0.2$ *and* $\alpha = 5°$.

In conclusion, the farfield points (selection I) is the most attractive point selection, since it yields good results and uses very few points at the same time. Figure 10.12 demonstrates that the pressure distribution is approximated well by the MPE model compared to the CFD solution obtained from TAU.



(a) The pressure distribution of the CFD solution.

(b) The pressure distribution of the MPE solution.

Figure 10.12: *The pressure distribution on the surface of the DLR-F12 configuration at* $M_\infty = 0.2$ *and* $\alpha = 5°$ *for the CFD reference solution and the MPE computed using only the farfield points and* $d = 2$ *modes.*

Computational time: Next, the computational time shall be investigated and compared to the CFD reference solution. For the computations an AMD Athlon$^{\mathrm{TM}}$ 64 X2 Dual Core Processor 4400+ with 2 GB RAM is used. The computer is operated under Debian GNU/LINUX.

Table 10.22 shows the time spend on computing the MPE solution and the reference solution. Here, the time for MPE is the average of 10 independent runs. The CFD computation is initialized with the freestream values and is stopped after 2.000 iterations. In this way the residual is dropped by seven orders of magnitude.

model	iterations (residual eval's)	time (in CPU s)
MPE (farfield points)	10	< 117.743
CFD	2000	15103.1

Table 10.22: *Computational time for the DLR-F12 configuration at* $M_\infty = 0.2$ *computed with the MPE using only the farfield points and* $d = 2$ *modes as well as with CFD. The time for MPE includes the computation of POD, but not the generation of the snapshots. Since MPE is not efficiently implemented yet, the time is expected to be less. Hence the* $<$ *sign.*

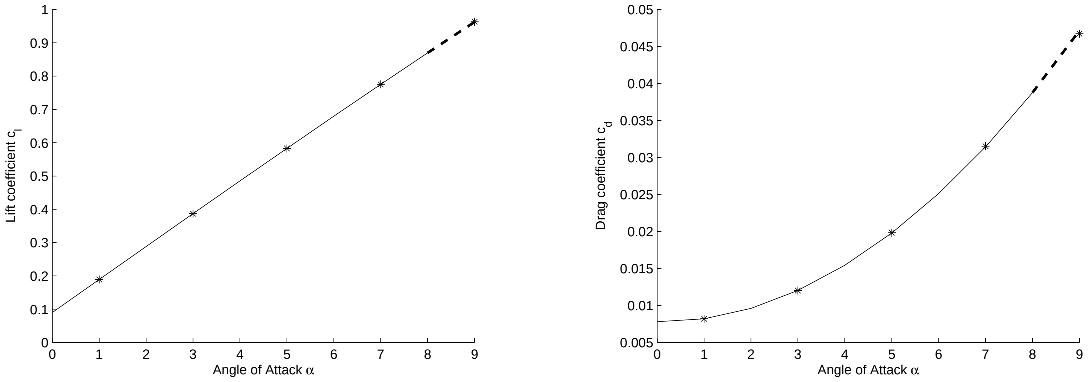
It can be seen that the time spend on computing the CFD solution is about 125 times more than the time spend on MPE. In addition, 200 times more iterations have to be computed. Again, the computational time is expected to be less, if the residual evaluations are computed in a sparse way. In fact, note that only 17.102 seconds are spend on reading the snapshots and computing the POD. Hence, there is a lot of room for improvement, if the residuals are computed sparsely.

Solutions at some intermediate angles of attack: In the following the solutions at some other intermediate angles of attack, namely $\alpha \in \{1°, 3°, 7°\}$, are computed. Again, the solution shall also be extrapolated at $\alpha = 9°$. Table 10.23 shows the aerodynamic coefficients computed with CFD and MPE as well as their errors.

AoA	c_l of MPE (error)	c_l of CFD	c_d of MPE (error)	c_d of CFD
1°	$1.893 \cdot 10^{-1}$ (0.0%)	$1.893 \cdot 10^{-1}$	$8.217 \cdot 10^{-3}$ (0.31%)	$8.192 \cdot 10^{-3}$
3°	$3.869 \cdot 10^{-1}$ (0.03%)	$3.868 \cdot 10^{-1}$	$1.201 \cdot 10^{-2}$ (0.25%)	$1.204 \cdot 10^{-2}$
5°	$5.827 \cdot 10^{-1}$ (0.0%)	$5.827 \cdot 10^{-1}$	$1.983 \cdot 10^{-2}$ (0.15%)	$1.980 \cdot 10^{-2}$
7°	$7.752 \cdot 10^{-1}$ (0.01%)	$7.752 \cdot 10^{-1}$	$3.153 \cdot 10^{-2}$ (0.29%)	$3.144 \cdot 10^{-2}$
9°	$9.631 \cdot 10^{-1}$ (0.05%)	$9.626 \cdot 10^{-1}$	$4.667 \cdot 10^{-2}$ (1.02%)	$4.715 \cdot 10^{-2}$

Table 10.23: *Lift and drag coefficients of the solutions for different angles of attack of the DLR-F12 configuration at $M_\infty = 0.2$ computed with MPE using all farfield points and $d = 2$ modes as well as with CFD.*

These results are also visualized in Figure 10.13, in which the lift and drag coefficient are plotted over the angle of attack. Note that the lift curve is quite linear and the relative error is very small. For the drag coefficient the error is significantly less than one percent for the interpolation range and about one percent for the extrapolation.



(a) The lift coefficient.

(b) The drag coefficient.

Figure 10.13: *The aerodynamic coefficients plotted over the angle of attack for the DLR-F12 configuration at $M_\infty = 0.2$. The line represents the coefficients of CFD solution and the asterisks represent those of the solution of MPE.*

Comparison with Thin Plate Spline (TPS) interpolation: Finally, the results are compared with TPS interpolation. Again, TPS interpolation yields good results for $\alpha = 3°$ and $\alpha = 5°$, while for the angles close to the boundary of the interpolation range, $\alpha = 1°$ and $\alpha = 7°$, the error is much higher. This is not the case for the MPE solutions, which once again underlines the good global approximation property of the MPE system.

Also note that the extrapolated solution is much better approximated with MPE compared to interpolation. Unlike before, the static stall region has not been reached at an angle of 9° and thus MPE is capable of yielding a good approximation. For TPS interpolation the drag error is unacceptably high.

As observed in MPE, better results are obtained in general when using two instead of all four modes. Note, however, that the difference is marginal.

	$d = 2$ modes		$d = 4$ modes	
AoA	c_l of TPS (error)	c_d of TPS (error)	c_l of TPS (error)	c_d of TPS (error)
1°	$1.891 \cdot 10^{-1}$ (0.11%)	$7.577 \cdot 10^{-3}$ (7.51%)	$1.891 \cdot 10^{-1}$ (0.11%)	$7.570 \cdot 10^{-3}$ (7.59%)
3°	$3.869 \cdot 10^{-1}$ (0.03%)	$1.203 \cdot 10^{-2}$ (0.08%)	$3.869 \cdot 10^{-1}$ (0.03%)	$1.205 \cdot 10^{-2}$ (0.08%)
5°	$5.826 \cdot 10^{-1}$ (0.02%)	$1.985 \cdot 10^{-2}$ (0.25%)	$5.827 \cdot 10^{-1}$ (0.00%)	$1.984 \cdot 10^{-2}$ (0.20%)
7°	$7.748 \cdot 10^{-1}$ (0.06%)	$3.088 \cdot 10^{-2}$ (1.78%)	$7.749 \cdot 10^{-1}$ (0.05%)	$3.084 \cdot 10^{-2}$ (1.91%)
9°	$9.654 \cdot 10^{-1}$ (0.29%)	$5.061 \cdot 10^{-2}$ (7.34%)	$9.654 \cdot 10^{-1}$ (0.29%)	$5.077 \cdot 10^{-2}$ (7.68%)

Table 10.24: *Lift and drag coefficients of the solutions for different angles of attack of the DLR-F12 configuration at $M_\infty = 0.2$ obtained with TPS interpolation using $d = 2$ and $d = 4$ modes.*

10.4 DLR-F15 airfoil for angles of attack near the static stall

In the following the same airfoil as in Section 10.1 is investigated. Again, the viscous flow at a freestream Mach number of $M_\infty = 0.22$, a Reynolds number of 10.549 million and a reference temperature of $118\,K$ is considered. But, unlike before, snapshots at higher angles of attack are used, namely $\alpha \in \{7°, 7.5°, 8°, 8.5°, 9°, 9.5°, 10°\}$. After constructing the reduced order model with these snapshots, our goal is to compute the flow at $\alpha = 8.75°$, which is near static stall. Note that only snapshots near static stall are used, since the snapshots further away from it exhibit different dynamics.

Snapshots are computed using the RANS solver Tau with the Spalart-Allmaras turbulence model.

Influence of the number of used modes: In the beginning, take a look at the number of modes used. Again, we first consider the relative energy content. It is given in Table 10.25. Note that the first mode carries about 99.9 % of the energy.

Next, we investigate the influence of the number of modes used in the MPE reduced order model. For this purpose, only the farfield points are considered in the residual evaluations of the MPE. Table 10.26 shows the lift and drag coefficient as well as their errors compared to the CFD reference solution.

The results of Table 10.26, however, are not conclusive. This is due to the fact that the errors do not decrease with an increasing number of modes used, as it would be expected. As a matter

j	1	2	3	4	5	6
$E(j)$	$9.9978 \cdot 10^{-1}$	$1.9584 \cdot 10^{-4}$	$1.3277 \cdot 10^{-5}$	$8.1977 \cdot 10^{-6}$	$4.2468 \cdot 10^{-7}$	$7.1714 \cdot 10^{-8}$

Table 10.25: *Relative energy content of the modes corresponding to the snapshots at* $\alpha \in \{7°, 7.5°, 8°, 8.5°, 9°, 9.5°, 10°\}$ *of the high-lift take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*).*

# of modes	residual evaluations	c_l (error in %)	c_d (error in %)
$d = 1$	7	2.807 (2.53%)	$6.796 \cdot 10^{-2}$ (2.10%)
$d = 2$	9	2.866 (0.49%)	$6.606 \cdot 10^{-2}$ (0.75%)
$d = 3$	10	2.868 (0.42%)	$6.581 \cdot 10^{-2}$ (1.13%)
$d = 4$	11	2.891 (0.38%)	$6.685 \cdot 10^{-2}$ (0.44%)
$d = 5$	21	2.926 (1.60%)	$6.720 \cdot 10^{-2}$ (0.96%)
$d = 6$	-	-	-
CFD	2000	2.880	$6.656 \cdot 10^{-2}$

Table 10.26: *Lift and drag coefficients of the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) computed with MPE, where the residual evaluations are restricted to the farfield points. The aerodynamic coefficients are predicted at* $\alpha = 8.75°$ *for different numbers of modes d and compared to the CFD reference solution at the same angle.*

of fact, using two modes yields a smaller error of the drag coefficient than using three modes and unacceptably high errors in the aerodynamic coefficients are observed when using five modes.

Note that when using six modes an unphysical value occurs in the residual evaluation of the CFD solver. As a result, the MPE method is stopped and does not converge. Consequently, no aerodynamic coefficients can be computed.

In the following we want to look at the modes. The corresponding coefficients are displayed in Table 10.27. It can be observed that only the first mode coefficient is similar in all computations. The second coefficient, however, increases, the more modes are chosen.

# of modes	1	2	3	4	5
$d = 1$	0.20093				
$d = 2$	0.19796	0.01158			
$d = 3$	0.19787	0.01212	-0.000569		
$d = 4$	0.19703	0.01471	0.002127	-0.002725	
$d = 5$	0.19549	0.01901	0.004412	-0.006022	-0.001223

Table 10.27: *The POD coefficients for the modes obtained with MPE, where the residual evaluations are restricted to the farfield points, using different numbers of modes d.*

In Figure 10.14 the energy modes are shown. Note that the second energy mode has negative values around the leading edge. Since the energy is coupled to the pressure, the increase of the second POD coefficient with the number of modes leads to a decrease in the drag. Furthermore, due to the negative values of the mode on the upper side of the surface, the lift is increased.

Note that – generally speaking – the third mode increases the drag, if multiplied with a positive coefficient, and decreases the drag otherwise. This partially explains the low c_d value when using three as opposed to four modes.

Nevertheless, it is desirable if the method would not compute the coefficients such that using more modes yields worse results. Note that this implies that the MPE method does not provide the optimal solution in the subspace. In the next chapter this will be analyzed.

Influence of the selection matrix: When considering only the farfield points, the use of four modes yields the best results. For this reason, we will investigate the influence of the selected points using four modes. In addition, we will also consider six modes, since for other point selections using all six modes might not provoke unphysical values in the residual evaluation and yield good results.

In the following the very same point selections as in the first test case, given in Table 10.4, will be used to analyze the influence of the selected points. Table 10.28 shows the aerodynamic coefficients.

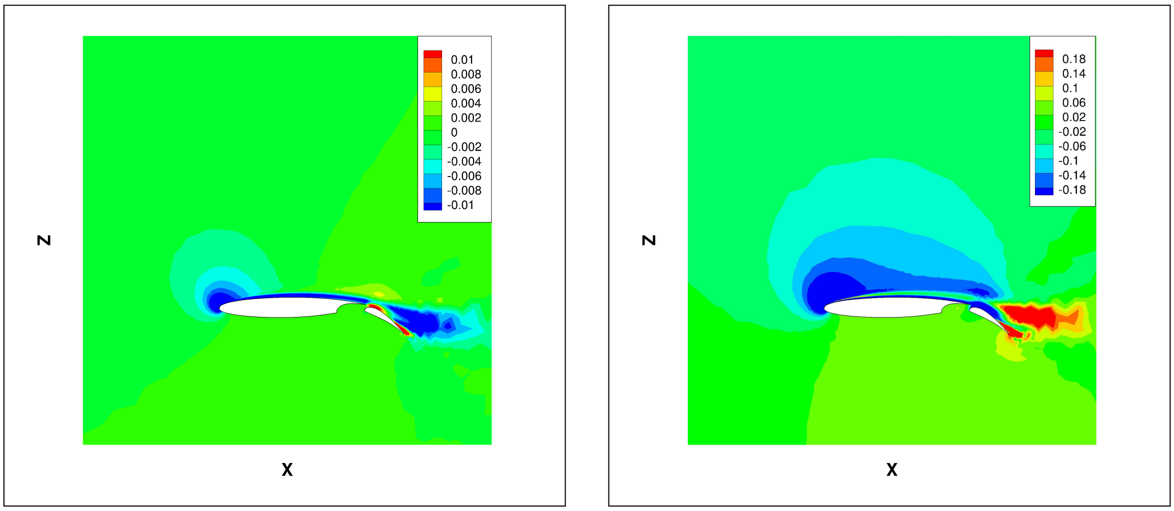
	$d = 4$ modes		$d = 6$ modes	
Selection	c_l (error)	c_d (error)	c_l (error)	c_d (error)
I	2.889 (0.31%)	$6.567 \cdot 10^{-2}$ (1.34%)	2.884 (0.14%)	$6.487 \cdot 10^{-2}$ (2.54%)
II	2.891 (0.38%)	$6.685 \cdot 10^{-2}$ (0.44%)	-	-
III	2.906 (0.90%)	$6.802 \cdot 10^{-2}$ (2.19%)	2.708 (5.97%)	$5.934 \cdot 10^{-2}$ (10.85%)
IV	2.879 (0.03%)	$6.384 \cdot 10^{-2}$ (4.09%)	2.880 (0.00%)	$6.645 \cdot 10^{-2}$ (0.17%)
V	2.889 (0.31%)	$6.567 \cdot 10^{-2}$ (1.34%)	2.884 (0.14%)	$6.487 \cdot 10^{-2}$ (2.54%)
VI	2.877 (0.10%)	$6.669 \cdot 10^{-2}$ (0.20%)	2.880 (0.00%)	$6.656 \cdot 10^{-2}$ (0.00%)

Table 10.28: *Lift and drag coefficients computed with MPE for different point selections for the DLR-F15 take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) at* $\alpha = 8.75°$*. Point selections I see Figure 10.1(b); selections II - V see Table 10.4.*

Overall, the worst results are obtained when using points in the boundary layer along with the farfield (selection III), since in this case the largest errors in the coefficients occur using both four as well as all six modes. The only exception is the error in the drag coefficient when using four modes, in which case selection IV yields an even greater error.

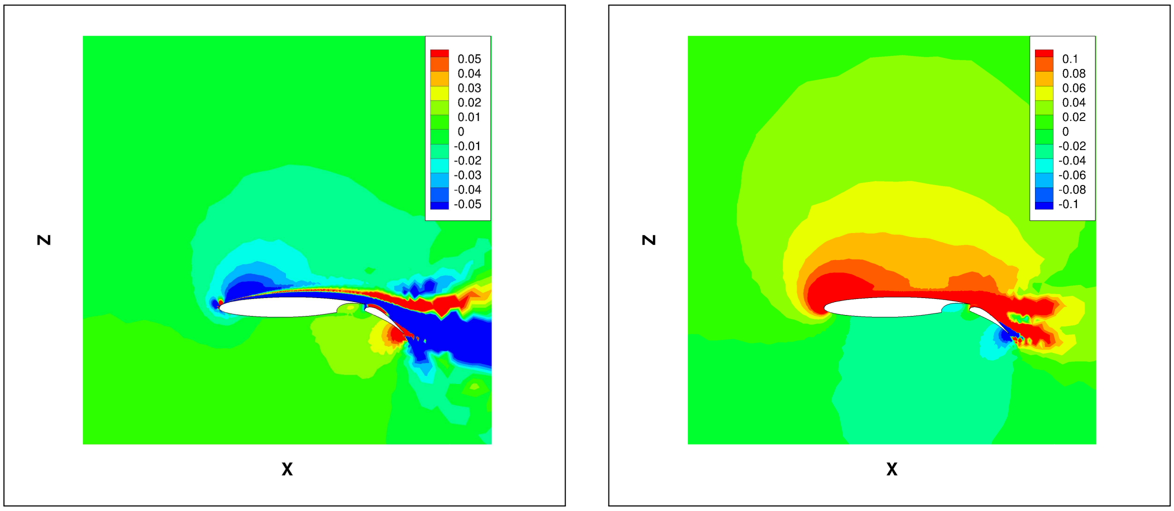
The best results are obtained when using all points and both four or all six modes. In fact, in the latter case the aerodynamic coefficients are approximated exactly.

When using six modes, the second best point selection is number IV, which comprises points in a region roughly adjacent to the boundary layer. With four modes, the solution of the farfield



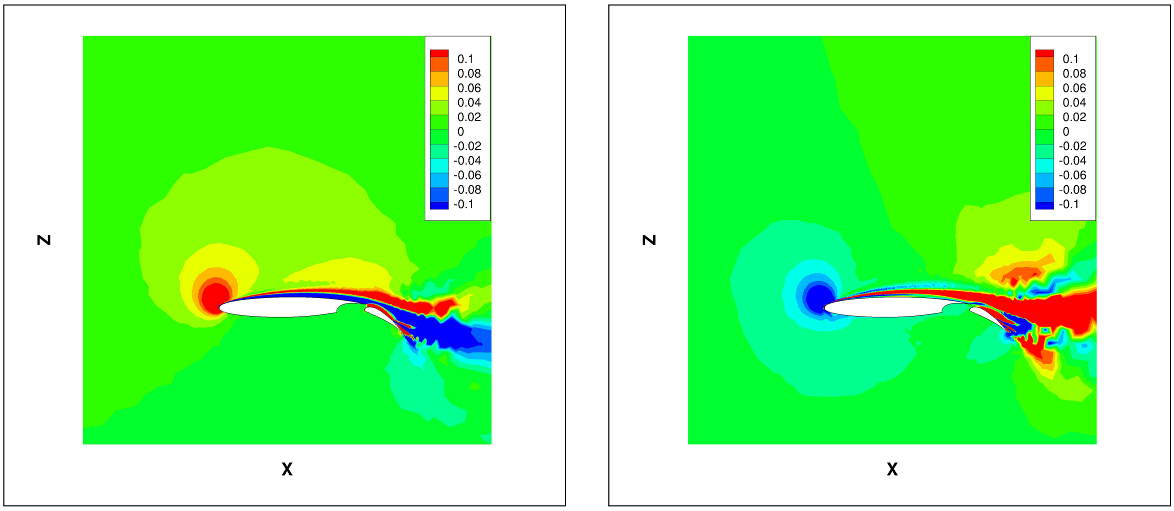
(a) Energy mode 1.

(b) Energy mode 2.



(c) Energy mode 3.

(d) Energy mode 4.

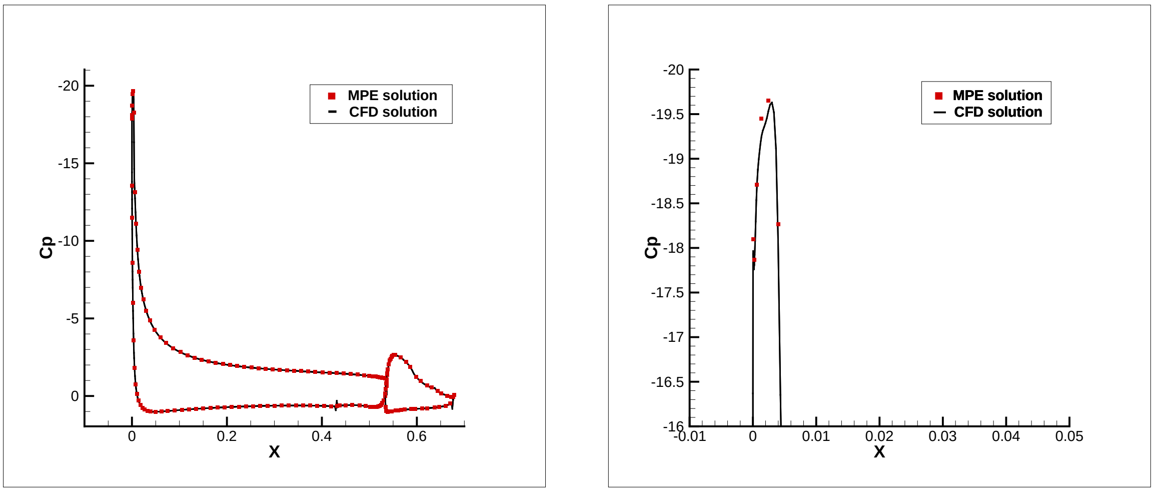


(e) Energy mode 5.

(f) Energy mode 6.

Figure 10.14: *The modes of the DLR-F15 take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) near static stall.*

point selection yields a good approximation of the drag coefficient and a reasonably well one of the lift coefficient. In Figure 10.15 the surface pressure distribution for this solution is shown. Note that only in the suction peak deviations from the CFD reference solution can be seen.



(a) Full view.

(b) Zoomed view of the suction peak.

Figure 10.15: *The pressure distribution on the surface of the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) at* $\alpha = 8.75°$ *for the MPE solution obtained using four modes and only farfield points in the residual evaluations.*

Computational time: Next, the computational time shall be the focus of discussion. For the computations an Intel® Core$^{\text{TM}}$ i7-3770 CPU @ 3.40GHz with 16 GB RAM is used. The computer is operated under Debian GNU/LINUX.

The computation of the CFD reference solution at $\alpha = 8.75°$ is initialized with the solution at $\alpha = 8.5°$. Convergence to steady state is detected, if the change in the aerodynamic coefficients is below a user-defined tolerance over some specified interval of iterations. The tolerances for the change in the lift c_l, drag c_d and pitching moment c_m coefficient are set to 10^{-3}, 10^{-5}, and 0.1, respectively. The interval is chosen to be 300 iterations. With these setting 2050 iterations are needed. Note that the residual itself is not part of the convergence criterion, but its norm is smaller than $5 \cdot 10^{-8}$ in the end.

Table 10.29 reveals the computational time spent on the CFD solution. With 885.6 CPU seconds, it is more than two hundred times slower than the time needed for the MPE computation, for which the residual evaluations are restricted to the farfield points. In addition, the number of iterations is larger than the number of residual evaluations of the MPE computation by a factor of approximately two hundred.

It is expected that the time spent on MPE is actually dramatically less than the given time due to the implementation. Note that only 0.656 CPU seconds are spent on reading the snapshots and computing the POD. This leaves plenty of time for improvement, as the overall time for MPE is 3.402 CPU seconds.

model	iterations (residual eval's)	time (in CPU s)
MPE (farfield points)	11	< 3.402
CFD	2050	885.6

Table 10.29: *Computational time for the take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) at* $\alpha = 8.75°$ *computed with the MPE using only the farfield points and with CFD. The time for MPE includes the computation of POD, but not the generation of the snapshots. Since MPE is not efficiently implemented yet, the time is expected to be less. Hence the* $<$ *sign.*

Comparison with Thin Plate Spline (TPS) interpolation: Finally, as a last test, we want to compare the MPE results to TPS interpolation. Note that in the latter method there is only a slight difference in the drag coefficient when using four as opposed to six modes.

When using all points in the residual evaluations of the MPE method, better approximations of c_l and c_d are obtained compared to TPS interpolation. This is also the case when using six modes and selection IV for MPE. Using the farfield points and four modes, however, yields slightly worse results.

All in all, MPE outperforms TPS interpolation, if the right point selection is chosen.

# of modes	c_l of TPS (error)	c_d of TPS (error)
$d = 4$	2.886 (0.21%)	$6.684 \cdot 10^{-2}$ (0.42%)
$d = 6$	2.886 (0.21%)	$6.685 \cdot 10^{-2}$ (0.44%)

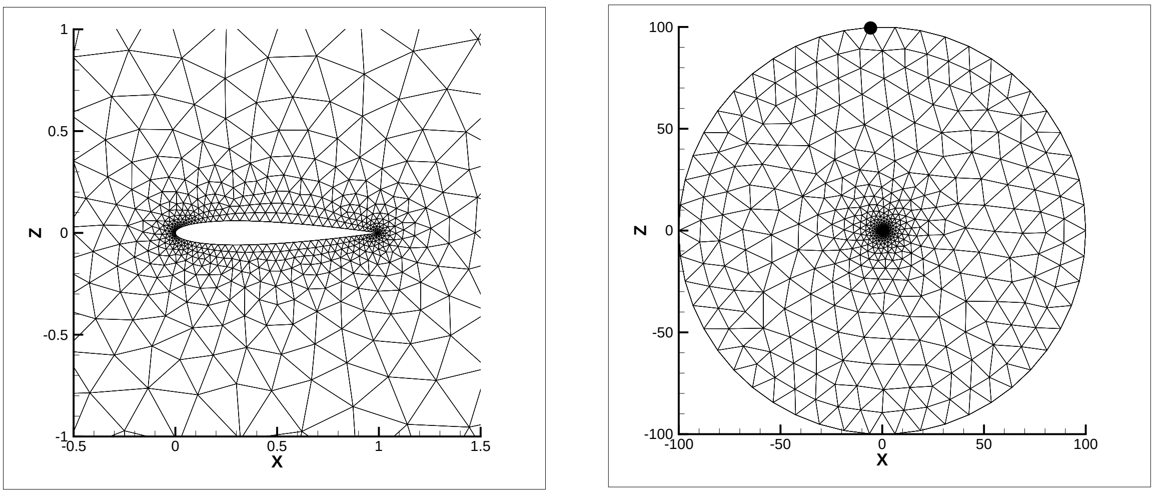
Table 10.30: *Lift and drag coefficients for the DLR-F15 take-off configuration (Re* $= 10.5$ *million,* $M_\infty = 0.22$*) at* $\alpha = 8.75°$ *obtained with TPS interpolation using* $d = 4$ *and* $d = 6$ *modes.*

10.5 NACA0012 airfoil at transonic speed

In the following a transonic flow is considered. Note that in the transonic flow regime shock waves occur, which pose a challenge to the reduced order models. In fact, in the following test case it will be observed that considering points lying on the shock for the residual evaluation of the MPE model has a negative influence on the resulting aerodynamic coefficients.

The flow past the NACA0012 airfoil at a transonic Mach number of $M_\infty = 0.73$ is considered. The computational domain is discretized with a coarse mesh of 1,140 grid points. It is shown in Figure 10.16. Note that the flow is modeled with the Euler equations. Due to the coarseness of the grid and the absence of a boundary layer, specific points can easily be illustrated, even those close to the surface. This is very helpful when studying the impact of the shock on the solution.

The snapshots are flow solutions at different angles of attack, namely $\alpha \in \{3°, 4°, 6°, 7°\}$. With the help of these snapshots, the flow at $\alpha = 5°$ shall be computed with the MPE reduced order model.



(a) Detailed view of the computational grid close to the surface.

(b) Full view with one marked point constituting point selection I of Table 10.34

Figure 10.16: *The computational grid of the NACA0012 airfoil with* $N = 1,140$ *points.*

Since the Euler equations are used to model the flow, it is assumed that the flow is inviscid. The modeling with the Euler equations is chosen, on the one hand, because it allows a coarse grid, but most importantly, because transonic flows modeled with the Navier-Stokes equations could not be solved accurately with an MPE reduced order model. In fact, the author has not been able to accurately simulate any other transonic flows with missing point estimation. It is expected that this is due to the fact that the occurring shocks pose a challenge to the MPE method.

Influence of the number of used modes: As before, we start by investigating the relative energy content, which is given in Table 10.31. Again, the first mode carries almost all the energy.

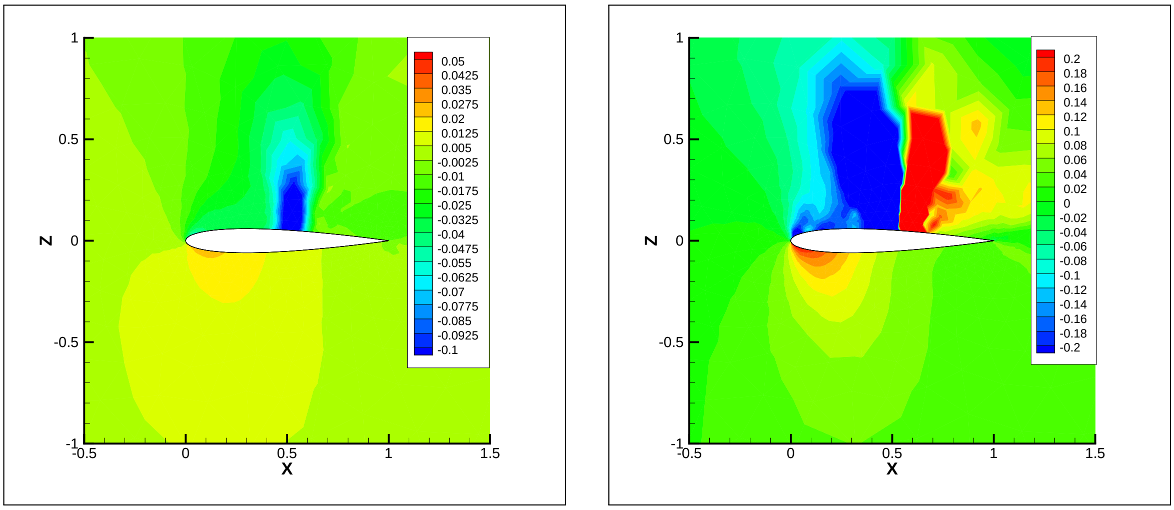
d	1	2	3	4
$E(d)$	$9.99874 \cdot 10^{-1}$	$1.15971 \cdot 10^{-4}$	$9.97872 \cdot 10^{-6}$	0

Table 10.31: *Relative energy content of the modes corresponding to the snapshots at* $\alpha \in \{3°, 4°, 6°, 7°\}$ *of the NACA0012 airfoil.*

Next, the number of modes used for the MPE model is varied, while keeping the points used in the residual evaluation fixed (using only the farfield points). Table 10.32 shows the aerodynamic coefficients for the solution at a Mach number of $M_\infty = 0.73$ and for $\alpha = 5°$. It can be seen that both the error in the lift and the drag coefficient decrease with an increasing number of modes used. This is both the expected as well as the desired behavior.

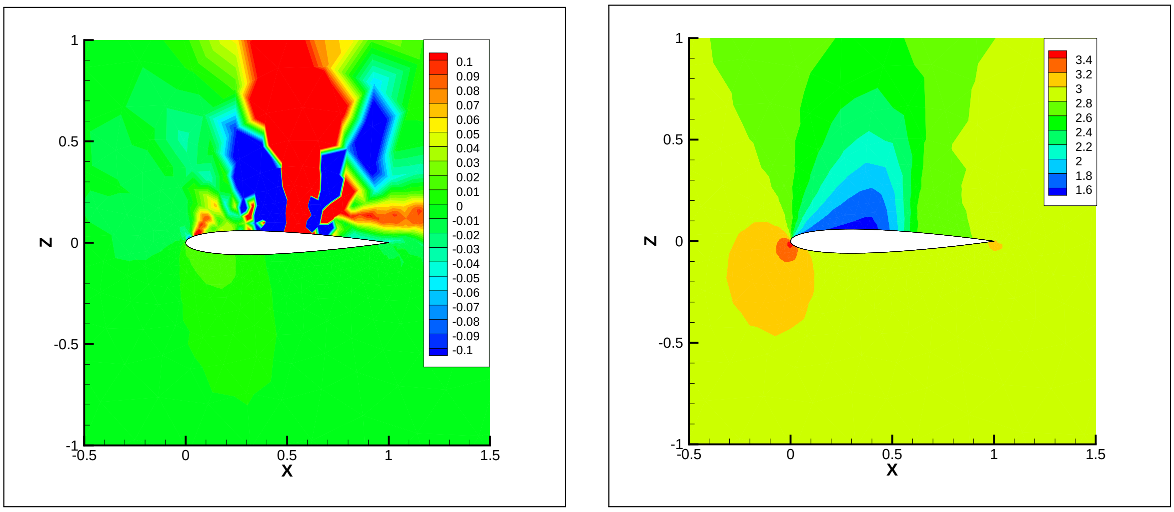
The modes are plotted in Figure 10.17. The corresponding coefficients can be found in Table 10.33. If only one mode is used, the POD coefficient is so small that the average dominates the solution (confer (7.7)).

When using two modes, the sign of the first POD coefficient changes. Note that the magnitude



(a) Energy mode 1.

(b) Energy mode 2.



(c) Energy mode 3.

(d) The average (mean flow).

Figure 10.17: *The energy modes and average of the NACA0012 airfoil test case.*

# of modes	residual evaluations	c_l (error in %)	c_d (error in %)
$d = 1$	6	$8.980 \cdot 10^{-1}$ (3.74%)	$6.713 \cdot 10^{-2}$ (3.53%)
$d = 2$	8	$9.272 \cdot 10^{-1}$ (0.61%)	$6.902 \cdot 10^{-2}$ (0.82%)
$d = 3$	9	$9.278 \cdot 10^{-1}$ (0.55%)	$6.928 \cdot 10^{-2}$ (0.45%)
CFD	168	$9.329 \cdot 10^{-1}$	$6.959 \cdot 10^{-2}$

Table 10.32: *Lift and drag coefficients of the NACA0012 ($M_\infty = 0.73$) computed with MPE, where the residual evaluations are restricted to the farfield points. The aerodynamic coefficients are predicted at $\alpha = 5°$ for different numbers of modes d and compared to the CFD reference solution at the same angle.*

# of modes	1	2	3
$d = 1$	0.00068073		
$d = 2$	-0.00298303	0.07302514	
$d = 3$	-0.00304721	0.07050316	-0.01586297

Table 10.33: *The POD coefficients for the modes obtained with MPE, where the residual evaluations are restricted to the farfield points, using different numbers of modes d.*

of the second mode is significantly larger, i.e. the contribution of the second mode dominates. When using three modes the coefficients of the first two modes change only slightly.

In Figure 10.17 it can be seen that the second mode contains a strong discontinuity, where the shock occurs. As a result, it is expected that the second mode is suited to approximate the shock. This is reflected in the fact that both the lift and drag are closer to the CFD reference solution, if two or more modes are used (see Table 10.32).

Influence of the selection matrix: Next, the influence of the points used in the residual evaluation of the MPE model is examined. Altogether, four different point selections are investigated: The first consist of only one single point in the farfield boundary, which is illustrated in Figure 10.16(b). The second consist of all farfield points. Finally, the last two selections comprise points that have a specific distance to the surface in addition to the farfield. In fact, selection III contains the points with a distance between 0.2 and 1.0 grid units from the surface and selection IV those points with a distance between 2.0 and 9.0 grid units from the surface. Table 10.34 summarizes the point selections.

In Table 10.35 the aerodynamic coefficients are shown for the different point selections. It can be seen that selection I, II, and IV yield approximately the same results. For point selection III, which consists of points close to the surface, the lift and drag coefficient are worse.

In order to investigate which of the points close to the surface have a negative influence on the aerodynamic coefficients, we utilize Algorithm 2. We initialize the algorithm with all farfield boundary points, that is, $\mathbb{X}$ is the set of all farfield points. Algorithm 2 then chooses in each

point selection	# of points	min distance	max distance
I (1 farfield point)	1	-	-
II (entire farfield)	50	99.0	101.0
III	221	0.2	1.0
IV	176	2.0	9.0

Table 10.34: *The definition of the point selections for the NACA0012 airfoil, where min/max distance stands for the minimal and maximal distance from the surface of the configuration in grid units (length of the airfoil = 1 grid units).*

point selection	residual eval's	c_l (error in %)	c_d (error in %)
I	10	$9.278 \cdot 10^{-1}$ (0.55%)	$6.927 \cdot 10^{-2}$ (0.46%)
II	9	$9.278 \cdot 10^{-1}$ (0.55%)	$6.928 \cdot 10^{-2}$ (0.45%)
III	15	$9.511 \cdot 10^{-1}$ (1.95%)	$7.109 \cdot 10^{-2}$ (2.16%)
IV	10	$9.277 \cdot 10^{-1}$ (0.56%)	$6.926 \cdot 10^{-2}$ (0.47%)
CFD	168	$9.329 \cdot 10^{-1}$	$6.959 \cdot 10^{-2}$

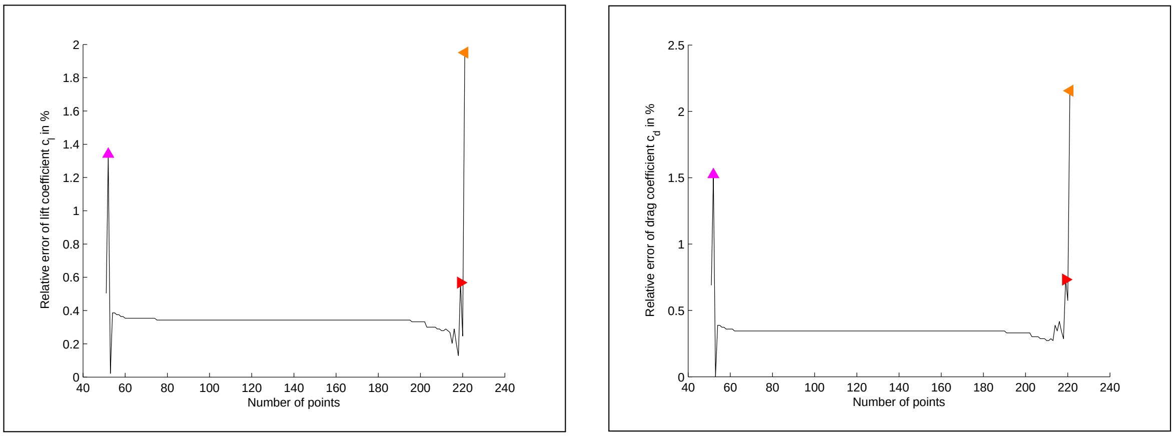
Table 10.35: *Lift and drag coefficients computed with MPE for different point selections for the NACA0012 airfoil ($M_\infty = 0.73$, $\alpha = 5°$). Point selections I see Figure 10.16(b); II - IV see Table 10.34*

iterative step the point having a distance between 0.2 and 1.0, which minimize the condition number of the matrix $U_d^T \tilde{\Omega} \tilde{P} \tilde{P}^T U_d$.

Figure 10.18 shows the lift and drag coefficient for each point selection, which is determined in the iterations of Algorithm 2. Note that in each step of the algorithm a new point is appended to the current point selection. The markers show those points, which – when appended to the selection – have a negative influence on the aerodynamic coefficient. These points are illustrated in Figure 10.19(a), which shows the pressure distribution of the TAU reference solution and the underlying grid. It can be seen that these points lie close to or even on the shock.

Note that the best approximation of the aerodynamic coefficients is achieved, after the algorithm has selected three points. These three points are marked in Figure 10.19(b), which along with the farfield establish point selection V. Obviously, they are also close to the shock region. Note that only the three points together yield good results, while each point alone does not. The aerodynamic coefficients as well as their errors are shown in Table 10.36. With this point selection MPE yields the exact drag coefficient and only an insignificantly small error in the lift coefficient.

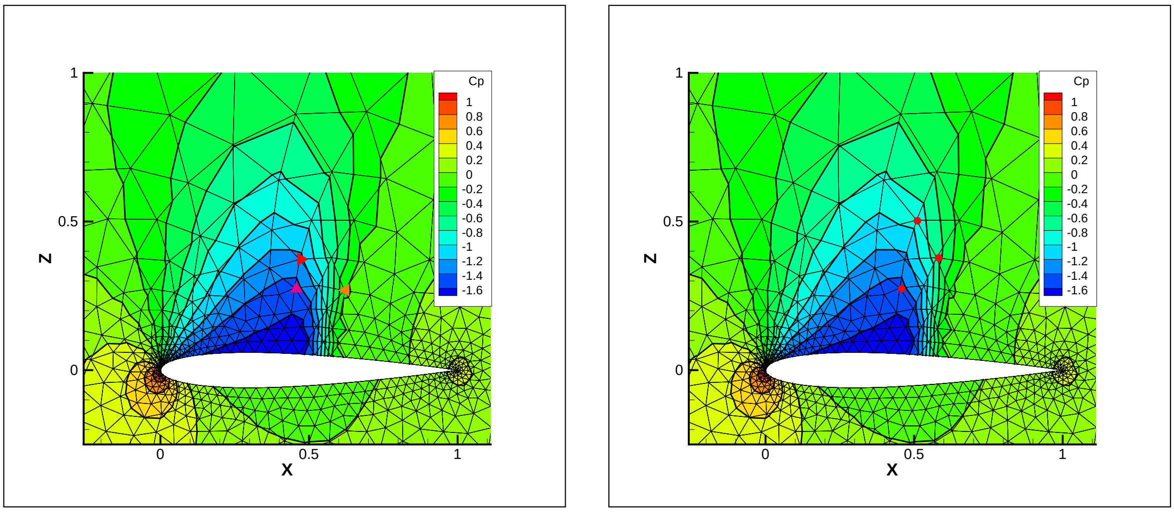
In Figure 10.20(b) the surface pressure distribution obtained with MPE using point selection V is visualized. It can be seen that it closely matches the pressure distribution of the CFD reference solution. Figure 10.20(a) shows the surface pressure distribution of the MPE solution when only the farfield points are used for the residual evaluations. Obviously, the shock is



(a) The error of the drag coefficient c_l.

(b) The error of the drag coefficient c_d.

Figure 10.18: *The error of the aerodynamic coefficients plotted over the number of points selected by Algorithm 2, where the set of indices* $\mathbb{I}$ *correspond to those points, which have a distance between 0.2 and 1.0 grid units. Note that the farfield points are also always included. The markers show those points illustrated in Figure 10.19(a), which are added in Algorithm 10.18 and have a negative influence on the aerodynamic coefficients.*



(a) The points, which have a negative influence on the aerodynamic coefficients in the test.

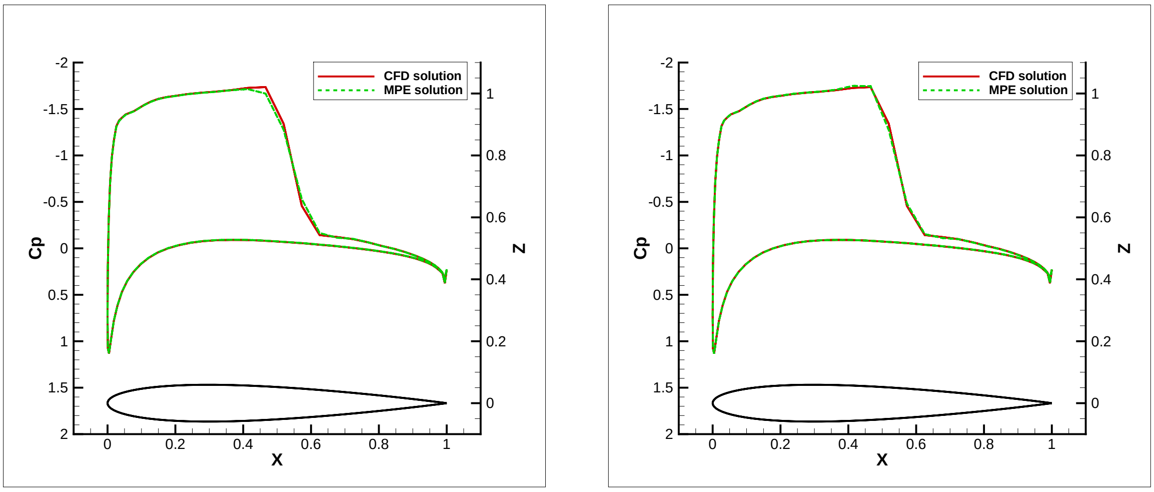
(b) Those points selected along with the farfield, which yield the best results in the test.

Figure 10.19: *Points playing an important role in the test using Algorithm 2.*

point selection	residual eval's	c_l (error in %)	c_d (error in %)
V	15	$9.331 \cdot 10^{-1}$ (0.02%)	$6.959 \cdot 10^{-2}$ (0.0%)
CFD	168	$9.329 \cdot 10^{-1}$	$6.959 \cdot 10^{-2}$

Table 10.36: *Lift and drag coefficients computed with MPE for point selection V for the NACA0012 airfoil ($M_\infty = 0.73$, $\alpha = 5°$). Point selections V consists of the farfield points and the three points visualized in Figure 10.19(b).*

smeared out. In fact, the pressure coefficient c_p drops further upstream of the shock compared to the CFD reference solution.



(a) The surface pressure distribution of the MPE solution considering only the farfield points in the residual evaluations.

(b) The surface pressure distribution of the MPE solution considering three points selected with Algorithm 2 as well as the farfield.

Figure 10.20: *The surface pressure distribution of two MPE solutions compared to the CFD reference solution.*

Since some of the points close to the region, where the shock occurs, have a negative influence on the aerodynamic coefficients and at the same time a combination of three shock region points yields good results, it can be concluded that the MPE method is very sensitive to the shock region. As a result it is very difficult to find a suitable point selection, which can accurately approximate the flow.

Computational time: In the following the computational time shall be addressed. For the computations an Intel® Core™ i7-3770 CPU @ 3.40GHz with 16 GB RAM is used. The computer is operated under Debian GNU/LINUX.

The CFD reference computation is initialized with the freestream values and it is stopped when the residual is below 10^{-7}. In this way, 168 pseudo-time iterations are computed. In Table 10.37 the time spent on the CFD computation is given. With one CPU second it is about eight times slower than the MPE reduced order model computation, where only the farfield points are used in the residual evaluations.

Unlike in the previous test cases in this chapter, for which speed-ups of more than one hundred are obtained, for this particular test case the number of grid points is comparatively small. As a result, the CFD solver does not need many iterations to converge and the speed-up is not as impressive.

As before, the MPE model is not implemented efficiently yet. That is, the residual is not evaluated sparsely, but for each and every grid point. Consequently, it is expected that the time spent on MPE can be dramatically reduced. Note that only 0.011 CPU seconds are spent on reading the snapshots and computing the POD. The rest of the overall 0.124 CPU seconds is spent on the finding of the root, which involves the residual computation.

Note that the computation of the three points with Algorithm 2 takes about 0.032 CPU seconds. Due to the fact that with point selection V fifteen residual evaluations are needed, the overall time is with 0.197 CPU seconds higher than that of selection II. Yet, this is still acceptable. Note, however, that if more points are selected with the algorithm, the time spent on the point selection quickly exceeds the time for the computation of the CFD reference solution.

model	iterations (residual eval's)	time (in CPU s)
MPE (farfield points)	9	< 0.124
CFD	168	1.0

Table 10.37: *Computational time for the NACA0012 airfoil ($M_\infty = 0.73$, $\alpha = 5°$) computed with the MPE using only the farfield points and with CFD. The time for MPE includes the computation of POD, but not the generation of the snapshots. Since MPE is not efficiently implemented yet, the time is expected to be less. Hence the $<$ sign.*

Comparison with Thin Plate Spline (TPS) interpolation: In the end, the results of the MPE solutions shall be compared to the TPS interpolation. Table 10.38 shows the lift and drag coefficient as well as their errors obtained with the latter method. It can be observed that the error of the interpolated solution decreases, the more modes are chosen. Note that the MPE conducted with point selection I, II, and IV approximate the aerodynamic coefficients slightly better than the interpolated solution. Point selection V, on the other hand, yields a significantly better approximation.

10.6 Summary

In the end, a brief summary of the numerical results shall be given: The missing point estimation reduced order model (9.13) has been used to obtain flow solutions for a three-dimensional configuration and various airfoils. In all tests, the angle of attack has been the only parameter to the system. Therefore, the snapshots were flow solutions at various angles of attack, while the Mach number was fixed. With the snapshots a POD basis is computed, which is inserted

# of modes	c_l (error in %)	c_d (error in %)
$d = 1$	$8.978 \cdot 10^{-1}$ (3.76%)	$6.712 \cdot 10^{-2}$ (3.55%)
$d = 2$	$9.268 \cdot 10^{-1}$ (0.65%)	$6.899 \cdot 10^{-2}$ (0.86%)
$d = 3$	$9.270 \cdot 10^{-1}$ (0.63%)	$6.903 \cdot 10^{-2}$ (0.80%)
CFD	$9.329 \cdot 10^{-1}$	$6.959 \cdot 10^{-2}$

Table 10.38: *Lift and drag coefficients of the NACA0012 ($M_\infty = 0.73$) obtained with TPS interpolation. The aerodynamic coefficients are predicted at $\alpha = 5°$ for different numbers of modes d and compared to the CFD reference solution at the same angle.*

into the reduced order model (9.13). This model is then used to obtain flow solutions at other angles of attack, which are not included in the snapshot set.

In this chapter five different test cases have been considered. Both a three-dimensional configuration as well as various two-dimensional airfoils have been used to evaluate the accuracy and the efficiency of the MPE method. In general, good results were obtained when using all modes and the entire farfield in the point selection matrix. An exception was the fourth test case, the DLR-F15 configuration near static stall, for which the use of the farfield along with all modes did not result in a converged solution. Four modes, however, yielded reasonable results.

Note that, in general, more modes did not always result in better approximations of the flow field. This implies that MPE does not find optimal coefficients for the POD representation.

Compared to CFD computations, the MPE reduced order model predicted a solution about one hundred times faster for almost all test cases. Only the NACA0012 airfoil, for which a coarse grid was employed, marks an exception. The comparison of MPE to TPS interpolation revealed that MPE outperforms interpolation for angles of attack, which lie near the border of the range of the snapshots.

While in this chapter, the angle of attack was the only parameter, in the next one the Mach number shall be used as an additional parameter as well. Furthermore, it shall be analyzed why the MPE does not necessarily find optimal coefficients for the POD representation.

Chapter 11

Analysis of the proposed method and its limitations

In Chapter 10 it has been shown that MPE works reasonably well for various test cases of a particular class of flow problems, namely, for predicting flows with the angle of attack as the only parameter to the reduced system. However, it has been observed that for some of the test cases the use of more modes resulted in worse approximations of the aerodynamic coefficients. As a consequence, it can be concluded that MPE does not find the optimal solution in the POD subspace.

In this chapter it shall be analyzed why the method fails to yield good results for the aerodynamic coefficients in some cases. To this end, different norms of the MPE and the full order residual are investigated.

Before these norms are analyzed for a test case, in which both the angle of attack and the Mach number are system parameters and for which the approximation of the aerodynamic coefficients are unacceptable, the norms are shown for two of the test cases of Chapter 10. Note that for these, MPE yields good results.

11.1 Investigating the norm of the residual of the MPE system

In this section the norm of the MPE residual is investigated. In particular, the 2-norm of the residual for each iterative solution of the MPE system given by $||W_d^T \boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})||_2$, where $\boldsymbol{a}$ is the vector of POD coefficients of the current iteration and $W_d^T = (U_d^T \tilde{\Omega} \tilde{P} \tilde{P}^T U_d)^{-1} U_d^T \tilde{P} \tilde{P}^T$, is compared to the 2-norm of the full order residual of this very same iterative solution, that is, $||\boldsymbol{R}(U_d \boldsymbol{a} + \bar{\boldsymbol{w}})||_2$. Ideally, both norms should approximately be of the same magnitude and should converge to zero in a comparable fashion.

In addition to the above norms, also the L_2-norm of the residual scaled by the inverse of the volumes is analyzed, i.e.

$$||\tilde{\Omega}^{-1}\boldsymbol{R}||_{L_2} = \sqrt{\left(\tilde{\Omega}^{-1}\boldsymbol{R}, \tilde{\Omega}^{-1}\boldsymbol{R}\right)_{\tilde{\Omega}}} = \sqrt{\boldsymbol{R}^T \tilde{\Omega}^{-1} \boldsymbol{R}} = ||\tilde{\Omega}^{-\frac{1}{2}} \boldsymbol{R}||_2. \tag{11.1}$$

Again, this is done for the iterative solutions of the MPE system given by $U_d \boldsymbol{a} + \bar{\boldsymbol{w}}$.

Note that the L_2-norm $\|\tilde{\Omega}^{-1}\boldsymbol{R}\|_{L_2}$ is equal to the 2-norm, where the residual is scaled by the square root of the inverse of the volumes (cf. (11.1)). This scaling puts more emphasis on the cells close to the surface, since these are much smaller in size than those close to the farfield. This is especially true for industrial configurations as considered in this thesis. Due to the fact that the aerodynamic coefficients c_l and c_d are determined by the forces on the surface, that is, c_l and c_d are functions of the pressure values at the surface, it is expected that the norm $\|\tilde{\Omega}^{-1}\boldsymbol{R}\|_{L_2}$ gives a better picture of how good the solutions approximate the aerodynamic coefficients and the full order solutions.

In the following subsections, the three norms shall be examined for two of the test cases considered in Chapter 10.

11.1.1 Two-dimensional DLR-F15 airfoil under take-off conditions

Consider again the first test case in Chapter 10, which is the section cut of the DLR-F15 under take-off conditions at a Mach number of $M_\infty = 0.22$. Again, the same snapshots as well as all four modes are used and the residual is evaluated only at the farfield points. As before, the solution at $\alpha = 7°$ is computed.

Figure 11.1 shows the three norms of the residual for each iteration step of MPE. Note that an iteration is illustrated in Figure 9.1 by the block named ”solve MPE system”. The initial solution is denoted by the zeroth iteration. It is depicted by ”determine start solution” in Figure 9.1. Note that in the first iteration non-physical values occurred in the residual and thus the norms could not be computed. Furthermore, note that the number of iterations differ from the number of residual evaluations, since the residual evaluations for building the Jacobian (illustrated by the block ”compute Jacobian” in Figure 9.1), are not considered as an iteration step.

In Figure 11.1 it can be seen that the 2-norm of the MPE residual $||W_d^T\boldsymbol{R}||_2$ and the 2-norm of the full order residual vector $||\boldsymbol{R}||_2$ of the initial, second and third iterative solution behave comparably. In addition, the two norms are approximately of the same magnitude for these iterations.

This changes in the fourth iteration, where the 2-norm of the MPE residual decreases significantly more than the 2-norm of the full order residual. Note also that in the last two iterations $||W_d^T\boldsymbol{R}||_2$ keeps decreasing with respect to the previous iteration, while $||\boldsymbol{R}||_2$ remains nearly constant. This indicates that in the last two iterations, the MPE method does not find solutions, which have lower residuals in sense of the full order model, but only in sense of the reduced order model. In fact, the residuals run into the null space of the projection or more precisely of the matrix W_d^T.

Obviously, this property of the MPE reduced order system is unwanted – yet expected – due to the low-order representation in the POD basis. For this test case, however, a nearly optimal solution is found as it can be seen in the aerodynamic coefficients (see Table 10.5) as well as the pressure distribution (see Figure 10.4) in Chapter 10. As a consequence, the fact that the residual runs into the null space does not mean that the solution is not approximated well. In fact, for all other point selection considered in Chapter 10 the residual runs into the null space

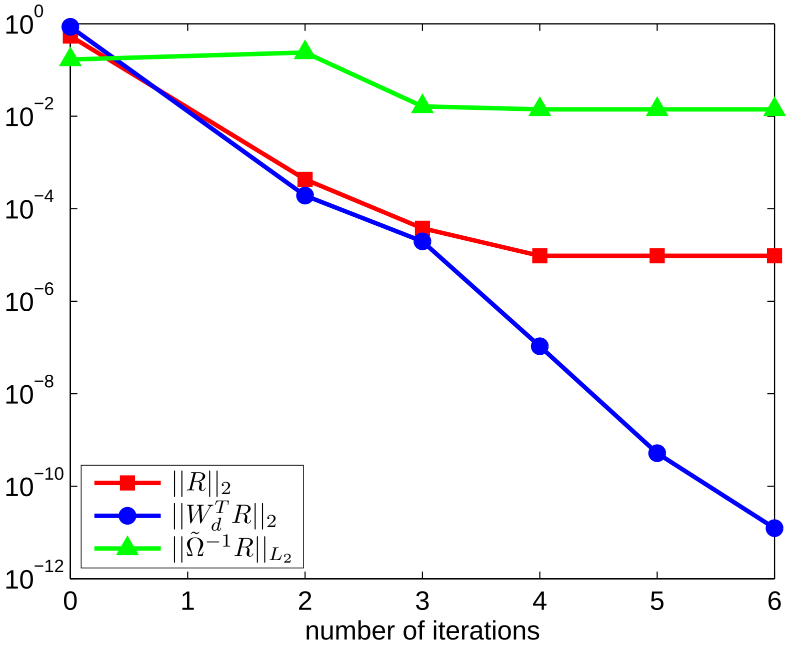


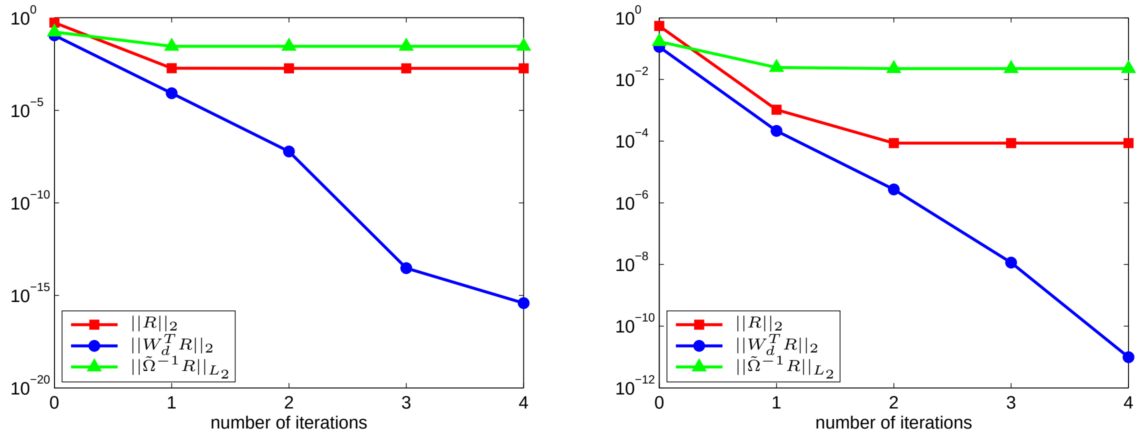
Figure 11.1: *DLR-F15 (Take-off conditions) test case: Norms of the full order residual and the MPE residual of the iterative solutions of the reduced order MPE system using four modes.*

as well.

In Figure 11.1, additionally, the L_2-norm of the residual vector scaled by the inverse of the volumes, i.e. $||\tilde{\Omega}^{-1}R||^2_{L_2}$, is depicted. Note that this norm behaves very differently from the other norms in the iterations. As a matter of fact, the L_2-norm is higher in the second iteration compared to that of the initial solution, while both other norms decrease. However, it can be noticed that this norm is smaller in the end than in the beginning and thus, a better solution also with respect to this norm is found.

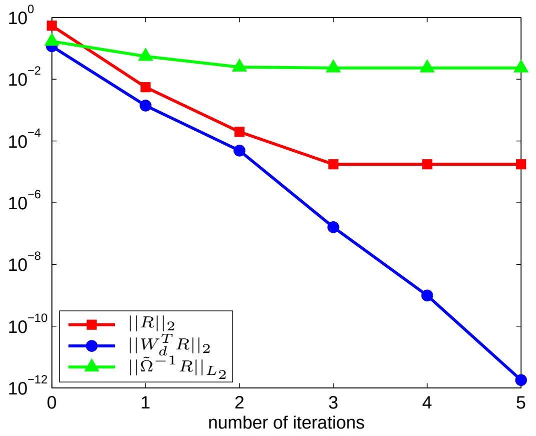
In the end, some comments on the behavior of the three norms shall be given, if less modes are used: First, we investigate the influence of the number of POD basis vectors (modes). In Figure 11.2 we see that the less modes are employed in the MPE computation, the greater are the final residual norms of $||\tilde{\Omega}^{-1}\boldsymbol{R}||^2_{L_2}$ and $||\boldsymbol{R}||_2$. This is not the case for the MPE residual norm $||W_d^T\boldsymbol{R}||_2$. In fact, the norm $||W_d^T\boldsymbol{R}||_2$ is smaller when using one mode or three modes compared to using four POD basis vectors. In the fact that the norms of $||W_d^T\boldsymbol{R}||_2$ are far below 10^{-10} it can be seen that the residual runs into the null space of the projection, no matter how many modes are used.

Furthermore, note that when utilizing one mode, only in the first iteration there is a similarity in the MPE residual norm and the 2-norm of the full order residual. In general, it seems that the less modes are used, the less similar these two norms behave.



(a) $d = 1$ mode

(b) $d = 2$ modes



(c) $d = 3$ modes

Figure 11.2: *DLR-F15 (Take-off conditions) test case: Norms of the full order residual and the MPE residual of the iterative solutions of the MPE system using different numbers of modes.*

11.1.2 Three-dimensional DLR F-12 configuration

As a second test case the three-dimensional DLR F-12 configuration from the previous chapter is considered. Again, the snapshots are the same as before, the two most relevant modes are used and the residuals are evaluated only at the farfield points. The computation of the subsonic flow at $(M_\infty, \alpha) = (0.2, 5°)$ with the help of MPE is considered.

Figure 11.3 shows the behavior of the three norms in the iteration process of MPE. Again, the 2-norm of the MPE residual, $||W_d^T \boldsymbol{R}||_2$, and the 2-norm of the full order residual vector, $||\boldsymbol{R}||_2$, are of the same magnitude and behave very similarly in the initial and in the first two iterations.

In the last three iterations $||W_d^T \boldsymbol{R}||_2$ rapidly decreases and is finally close to zero, while $||\boldsymbol{R}||_2$ is only slightly decreasing and eventually in the order of 10^{-5}. Therefore, the residual of the solution of the MPE system runs into the null space of the projection. This, however, does not seem to be a problem for this test case, since the solution yields good approximations to the aerodynamic coefficients (see Table 10.21) and the pressure distribution on the surface (Figure

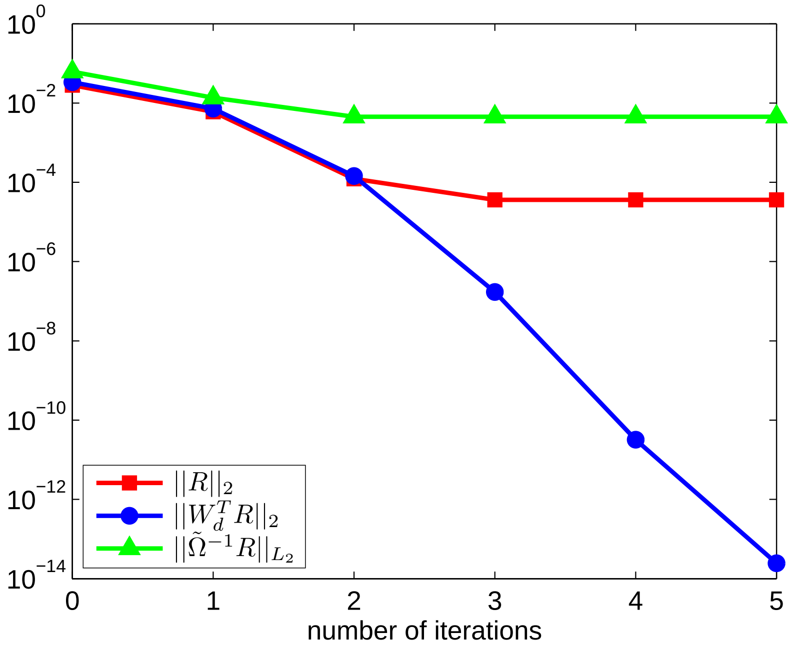


Figure 11.3: *DLR-F12 test case: Norms of the full order residual and the MPE residual of the reduced order system's iterative solutions, where two POD modes were used.*

10.12).

Note that the L_2-norm, $||\tilde{\Omega}^{-1}R||^2_{L_2}$, drops in the first two iterations and remains nearly constant afterwards. Although the L_2-norm is approximately of the same magnitude as the other norms in the initial solution and the first iteration, in general, it behaves less like the MPE residual norm than the 2-norm $||\boldsymbol{R}||_2$ for this test case.

11.1.3 DLR-F15 airfoil with the Mach number as an additional system parameter

In the previous two subsections, the residual norms of some of the test cases presented in Chapter 10 have been analyzed. These test cases belong to the problem class of predicting flows with the angle of attack as the only system parameter. In the test case considered in this subsection not only the angle of attack, but also the freestream Mach number is a varying parameter to the system. Hence, unlike in the previous test cases, here the snapshots are computed for different freestream Mach numbers as well as different angles of attack.

In particular, steady state snapshots for $\boldsymbol{w}(t = \infty; \alpha, M_\infty)$ are used, where $(\alpha, M_\infty) \in \{0°, 1°, 2°, 3°, 4°, 5°, 6°, 7°\} \times \{0.22, 0.24\}$. Thus, altogether sixteen snapshots are computed. Note that the same computational grid as in Figure 10.1 is used.

At first, the number of modes is determined. Having computed the POD, take a look at the energy content of the modes. In Table 11.1 these values are presented.

Note that unlike before, where only α was a system parameter, here not only the first but also the second mode carries a lot of energy. This means that the flow is more complicated

j	1	2	3	4
$E(j)$	$5.6152 \cdot 10^{-1}$	$4.3847 \cdot 10^{-1}$	$1.1072 \cdot 10^{-5}$	$2.6121 \cdot 10^{-6}$

Table 11.1: *Relative energy content of the modes corresponding to the snapshots at* $(\alpha, M_\infty) \in \{0°, 1°, 2°, 3°, 4°, 5°, 6°, 7°\} \times \{0.22, 0.24\}$ *of the DLR F15 configuration.*

than before and needs at least another mode to capture the relevant information of the flow. In this test case, four modes are chosen such that the relative energy content of these modes are 99.9999%.

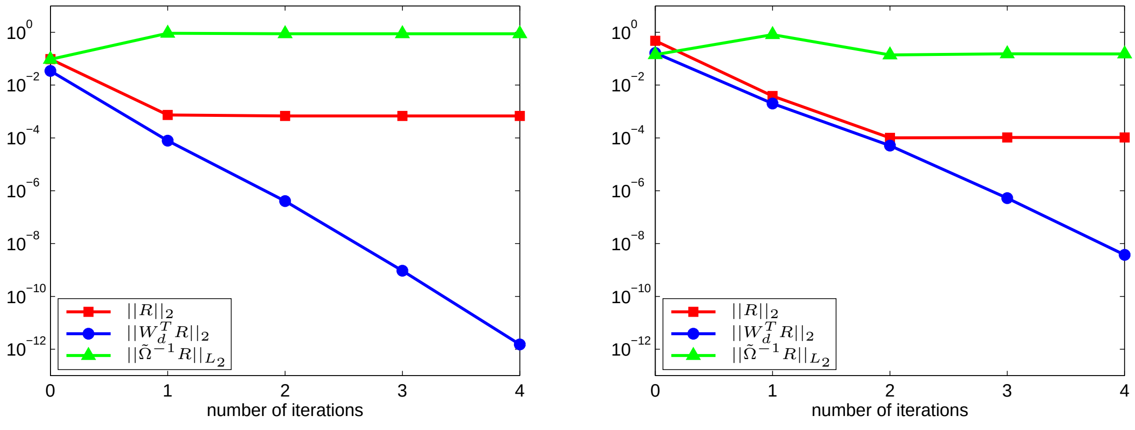
In the following MPE is used to compute the flows at two different flow conditions, namely at $(M_\infty, \alpha) = (0.23, 4°)$ and $(M_\infty, \alpha) = (0.23, 6°)$. As in the other test cases, only the farfield points are selected for the residual evaluations in the MPE.

In Table 11.2 the aerodynamic coefficients for the MPE as well as the CFD reference solution are shown. Note that the relative error of the predicted drag coefficient is higher than 20% for both angles of attack. Obviously, both solutions do not approximate the full order solution well.

AoA	c_l of MPE (error)	c_l of CFD	c_d of MPE (error)	c_d of CFD
4°	2.281 (5.00%)	2.401	$5.187 \cdot 10^{-2}$ (29.38%)	$4.009 \cdot 10^{-2}$
6°	2.572 (0.66%)	2.589	$5.516 \cdot 10^{-2}$ (21.90%)	$4.525 \cdot 10^{-2}$

Table 11.2: *Lift and drag coefficients of the solutions at* $M_\infty = 0.23$ *and* $\alpha = 4°$ *as well as* $\alpha = 6°$ *of the DLR F15 configuration computed with MPE using all farfield points and* $d = 4$ *modes.*

In order to find a reason for the bad approximation quality, take a look at the norms considered earlier. Figure 11.4 shows them for each iteration of the MPE system.



(a) $\alpha = 4°$ (b) $\alpha = 6°$

Figure 11.4: *DLR-F15 test case, where both* α *and* M_∞ *are system parameters: Norms of the full order residual and the MPE residual of the iterative solutions of the MPE system.*

For $\alpha = 6°$ it can be seen in Figure 11.4(a) that the 2-norm of the full order residual and of the MPE residual behave similarly and are of the same magnitude for the initial solution and the first two iterations. Then the full order residual norm remains constant, while the MPE residual norm keeps on decreasing. This suggests that the full order residual runs into the null space of W_d^T.

Note that unlike for the previous two test cases in this chapter, for this one the fact that the residual runs into the null space of W_d^T constitutes a problem. This is due to the fact that the found solution solves the MPE reduced order system, as it can be seen in the final MPE residual norm being below 10^{-8}, yet the very same solution does not approximate the aerodynamic coefficients well. As a matter of fact, the MPE finds a solution to the reduced order model, which is not a solution to the full order model.

This is reflected in the behavior of the L_2-norm, which is very different to that of the other norms. In fact, it increases in the first iteration, then drops in the second and remains nearly constant afterwards. Note that the L_2-norm is actually slightly higher in the end than for the initial solution.

Figure 11.4(a) shows the norms for $\alpha = 4°$. Note that here only in the first iteration the 2-norm of the full order residual and the MPE residual behave comparably. In all other iterations the latter norm decreases monotonously, while $||\boldsymbol{R}||_2$ remains nearly constant. Interestingly, the L_2-norm – unlike the 2-norm of the full order residual – increases in the first iteration and then is nearly constant. Recall that the L_2-norm has a bias towards the cells (with small volume) close to the surface. Thus the different behavior of the two norms, $||\boldsymbol{R}||_2$ and $||\tilde{\Omega}^{-1}\boldsymbol{R}||_{L_2} = ||\tilde{\Omega}^{-\frac{1}{2}}\boldsymbol{R}||_2$, means that the residual is decreased in the region close to the farfield, but is increased in close proximity to the surface. In other words, there is only a weak spatial correlation in the POD subspace between the farfield and the region close to the surface.

This suggests that another point selection should be used. In Table 11.3 the aerodynamic coefficients are presented for the point selections of Table 10.4. Note that only point selection VI, which consists of all points, yields somewhat acceptable results for the drag as well as the lift coefficient. With all other point selections the error in c_d is higher than ten percent for $\alpha = 4°$ and about twenty percent for $\alpha = 6°$.

Comparison with Thin Plate Spline (TPS) interpolation: Finally, we want to investigate the TPS interpolation and compare the results to those obtained with MPE. In Table 11.4 the aerodynamic coefficients as well as their errors are presented for both angles. It can be seen that for $\alpha = 4°$ the TPS yields good approximations of the coefficients. Note that $(M_\infty, \alpha) = (0.23, 4°)$ does not lie close to the boundary of the range of the snapshots. For $(M_\infty, \alpha) = (0.23, 6°)$, for which this is the case, the aerodynamic coefficients are not approximated as well. As a matter of fact, MPE using point selection VI yields better results. This behavior of TPS interpolation has been observed before in the other test cases.

	$\alpha = 4°$		$\alpha = 6°$	
Selection	c_l (error in %)	c_d (error in %)	c_l (error in %)	c_d (error in %)
I	2.338 (2.62%)	$4.726 \cdot 10^{-2}$ (17.88%)	-	-
II	2.281 (5.00%)	$5.187 \cdot 10^{-2}$ (29.38%)	2.572 (0.66%)	$5.516 \cdot 10^{-2}$ (21.90%)
III	2.364 (1.54%)	$4.741 \cdot 10^{-2}$ (18.26%)	2.587 (0.08%)	$5.478 \cdot 10^{-2}$ (21.06%)
IV	2.406 (0.21%)	$4.491 \cdot 10^{-2}$ (12.02%)	2.592 (0.12%)	$5.389 \cdot 10^{-2}$ (19.09%)
V	2.431 (1.25%)	$4.424 \cdot 10^{-2}$ (10.35%)	2.574 (0.58%)	$5.467 \cdot 10^{-2}$ (20.82%)
VI	2.408 (0.29%)	$4.206 \cdot 10^{-2}$ (4.91%)	2.600 (0.42%)	$4.635 \cdot 10^{-2}$ (2.43%)
CFD	2.401	$4.009 \cdot 10^{-2}$	2.589	$4.525 \cdot 10^{-2}$

Table 11.3: *Lift and drag coefficients computed with MPE using the point selections of Table 10.4 and* $d = 4$ *modes for the DLR-F15 test case for the computation of the flow at* $(M_\infty, \alpha) = (0.23, 4°)$ *as well as* $(M_\infty, \alpha) = (0.23, 6°)$*.*

AoA	c_l of TPS (error)	c_l of CFD	c_d of TPS (error)	c_d of CFD
4°	2.398 (0.12%)	2.401	$4.030 \cdot 10^{-2}$ (0.52%)	$4.009 \cdot 10^{-2}$
6°	2.611 (0.85%)	2.589	$4.830 \cdot 10^{-2}$ (6.74%)	$4.525 \cdot 10^{-2}$

Table 11.4: *Lift and drag coefficients computed with TPS interpolation for the DLR-F15 test case for the computation of the flow at* $(M_\infty, \alpha) = (0.23, 4°)$ *as well as* $(M_\infty, \alpha) = (0.23, 6°)$*.*

11.2 Summary

In this chapter, we have analyzed three different norms for the residual. These norms are the 2-norm of the MPE residual, the 2-norm of the full order residual, and finally the L_2-norm of the full order residual scaled by the inverse of the volumes. Comparison of the first to the second norm revealed for two test cases of Chapter 10 that the residual of the iterative solutions of the MPE reduced order model run into the null space of the projection matrix W_d^T. Nevertheless, good results for the aerodynamic coefficients and the pressure distribution on the surface are obtained, as we have seen in Chapter 10.

For another test case presented in this chapter, in which both the angle of attack and the Mach number are parameters to the reduced order system, this is not the case. In fact, the error in the drag coefficient for two different considered parameter settings was higher than 20 %. This is albeit the fact that the final MPE residual norm is below machine precision, i.e. 10^{-8}. Hence, the residual runs into the null space of the projection. In this case, this seems to pose a problem, since an acceptable approximation of the flow is found.

As a result, it shall be concluded that missing point estimation should not be used with the Mach number as a parameter. Rather, MPE should be employed to predict subsonic flows with the angle of attack as the only parameter.

Chapter 12

Closing

In this thesis an efficient reduced order model for the governing equations of computational fluid dynamics has been constructed. The nonlinearity of these equations poses a challenge for reduced order models. Existing projection-based methods have either not been efficient in the sense that they are independent of the full order (subspace projection) or have mostly not been applied to industrial test cases (Galerkin projection).

The method proposed in this work is referred to as missing point estimation. It alleviates both of the above difficulties in reduced order models for CFD. In fact, since the reduced order modeling is carried out on the discrete governing equations it is in this way similar to the subspace projection method. As a result, it inherits its strengths such as flexibility in the use of turbulence models and ease in the applicability to industrial test cases. But at the same time, it alleviates its weakness, which lies in the dependence on the full order. This is due to the way the reduced order model is set up. As a matter of fact, in the subspace projection method the right hand side of the governing equations has to be evaluated at each and every point. This is different for the missing point estimation, for which it has to be evaluated only at some selected points. This obviously speeds up the computation.

As it has been outlined in Chapter 9 the missing point estimation has been constructed such that the residual evaluation is carried out by a CFD solver, e.g. TAU. In this way all capabilities of the CFD solver can be utilized.

In Chapter 10 it has been shown that the proposed method, the missing point estimation, is capable of efficiently computing flows, where only the angle of attack is a system parameter. The MPE has been tested both for the Navier-Stokes as well as the Euler equations and for industrially relevant two-element high-lift airfoils as well as complex three-dimensional configurations.

For the particular problem class of predicting flows with α as a system parameter only the points on the farfield boundary sufficed to obtain accurate representations of the flow field with respect to the pressure distribution on the surface and the aerodynamic coefficients. This has been presented for subsonic flows within and outside of the static stall region as well as for an inviscid, transonic flow. Although other point selections could be found, which produced better results, especially for the transonic test case, the point selection consisting of only the farfield points provided overall good results and this point selection can easily be constructed. Due to these reasons, it is the most practical point selection for this problem class.

Comparison to TPS interpolation has led to the conclusion that MPE is able to obtain good solutions for all considered angles of attack in the interpolation range, while this is not true for TPS interpolation. In fact, the interpolation of angles close to the boundary of the interpolation range were not approximated well with errors in the drag coefficient up to three percent.

Besides angles in the interpolation range, for each of the first three test cases in Chapter 10, one angle of attack is considered, which lies outside of the range of the snapshots. This is referred to as extrapolation. In the first two test cases the extrapolated angle lies close to the static stall. As this behavior is not covered in the snapshots, the aerodynamic coefficients were not approximated as well as expected. Nevertheless, TPS interpolation performed reasonably well in the first test case. In the third test, the DLR-F12 configuration, the extrapolated angle is not close to the static stall and MPE dramatically outperformed the TPS interpolation.

For all test cases except the NACA0012, which has a significantly smaller number of grid points, the solutions are computed about one hundred times faster than with the original model. Note that this is mainly due to the small number of residual evaluations needed for solving the MPE reduced order model. As a matter of fact, the MPE has not been efficiently implemented yet. That is, the residuals are evaluated at each grid point, instead of computing it at only the needed locations, and the filtering is conducted a posteriori. It is expected that an efficient implementation will speed up the computations tremendously.

In all test examples of Chapter 10 the first mode of the POD basis carries almost all energy in the L_2 sense. Yet, generally speaking the more modes are taken into account, the better are the results. However, exceptions exist such as the three-dimensional DLR-F12 test case, for which using two modes yielded slightly better results than using all four modes.

For the DLR-F15 two-element high-lift airfoil within the static stall region using five modes yielded worse aerodynamic coefficients than using two or four modes. Note that this behavior of the method is both unexpected and undesirable. This leads to the conclusion that MPE does not provide an optimal solution in the POD subspace and, in fact, is not robust in the sense that it provides good results for all test cases and settings.

This is supported by the fact that the transonic flow presented in Chapter 10 of this thesis has been the only transonic test case, which could be set up by the author to yield reasonable results.

In order to analyze why the method fails, a second problem class is considered, for which both the angle of attack and the Mach number are parameters of the MPE system. For this problem class, a two-element section cut of the wing of the DLR-F15 configuration is considered. The relative error in the drag coefficient is larger than 20 % for the computation of two different angles of attack, where only the farfield points are used in the residual evaluations of the MPE reduced order model.

As a means to analyze why the method does not yield reasonable results, three different norms are investigated: First of all, the 2-norm of the MPE residual given by $||W_d^T \boldsymbol{R}||_2$ is considered for each iterative solution. In all the presented test cases, this norm is at least below the square root of machine precision, e.g. below 10^{-8}. This is, of course, expected as the MPE reduced order model tries to find the root of the projected residual $W_d^T \boldsymbol{R}$.

Secondly, the 2-norm of the full order residual $||\boldsymbol{R}||_2$ is studied. This norm drops along with the MPE residual norm in the first few iterations, but then actually remains nearly constant. The comparison of the two norms implies that the residual runs into the null space of the projection, since $||W_d^T \boldsymbol{R}||_2$ continuous decreasing, while $||\boldsymbol{R}||_2$ is not.

Finally, the L_2-norm of the full order residual scaled by the inverse of the volumes $||\tilde{\Omega}^{-1}\boldsymbol{R}||_{L_2}$ is considered. This norm is equivalent to the 2-norm of the residual scaled by the square root of the inverse of the volumes, i.e. $||\tilde{\Omega}^{-\frac{1}{2}}\boldsymbol{R}||_2$. Due to the fact that the volumes close to the surface are tremendously smaller than in the farfield, the scaling by the volumes puts a greater emphasis on the cells close to the surface.

In the second problem class, where both the angle of attack and the Mach number are system parameters, the L_2-norm of the full order residual scaled by the inverse of the volumes actually increases. On the one hand, this explains why the relative error in the drag is unacceptably high. But more importantly, this shows that although the MPE reduced order model is achieving its task to find a root of the projected residual, it fails to yield an acceptable approximation of the flow at the same time. This leads to the conclusion that projecting the governing equations onto a low-dimensional POD subspace is not advisable for all problem classes and test cases.

However, MPE yields good results for the test cases presented in Chapter 10. As a consequence, it is recommended to employ missing point estimation for predicting subsonic flows with the angle of attack as the only parameter. In fact, it should be preferred to TPS interpolation, as MPE outperforms TPS interpolation in a global sense. Note that even flows around very complex geometries like two-element high-lift airfoils as well as three-dimensional configurations can be approximated accurately at dramatically reduced computational times compared to CFD computations.

Future work: Although the source code for the proposed method is mainly written in the programming languages C and python, there is room for improvement in terms of efficiency. First of all, the sparse residual evaluation for only those points, which are selected, is not efficiently implemented yet. In fact, the residual is computed at each and every point, but only some selected components of the residual are used. Obviously, this can be improved, but an efficient implementation for the TAU is technically very challenging due to the fact that some parts of the CFD solver such as the preprocessing step have to be rewritten.

Another chance of improving the efficiency is the parallelization of the source code. In fact, some parts are already available in parallel such as reading the snapshot, the residual evaluation in TAU, and the POD basis computation. Computing a solution in parallel is also easy to implement, since it is a matrix-vector product of the POD basis matrix and the vector of POD coefficients. Note, however, that the parallelization of the source code has not been a goal in this work.

Further future work could also be the application of the missing point estimation to unsteady computations. Obviously, some adjustments such as implementing a time discretization scheme have to be made, but in general it is expected that the proposed method also works for unsteady flow problems.

Bibliography

[1] A. Antoulas, *Approximation of Large-Scale Dynamical Systems*, Advances in Design and Control, SIAM, Philadelphia, 2005.

[2] P. Astrid, *Fast reduced order modeling technique for large scale ltv systems*, in Proceedings of the American Control Conference, vol. 1, July 2004, pp. 762 –767.

[3] P. Astrid, *Reduction of process simulation models: A proper orthogonal decomposition approach*, PhD thesis, Technische Universiteit Eindhoven, 2004.

[4] P. Astrid and A. Verhoeven, *Application of Least Squares MPE technique in the reduced order modeling of electrical circuits*, in Proceedings of the 17th Int. Symp. MTNS, 2006, pp. 1980–1986.

[5] P. Astrid, S. Weiland, K. Willcox, and T. Backx, *Missing point estimation in models described by proper orthogonal decomposition*, IEEE Transactions on Automatic Control, 53 (2008), pp. 2237–2251.

[6] K. Atkinson and W. Han, *Theoretical Numerical Analysis: A Functional Analysis Framework (Texts in Applied Mathematics)*, Springer, 3rd ed., 2009.

[7] G. Batchelor, *An Introduction to Fluid Mechanics*, Cambridge University Press, Cambridge, 1967.

[8] D. Beazley, *Using SWIG to Control, Prototype, and Debug C Programs with Python*, in Proc. 4th Int'l Python Conf., IOS Press, 1996.

[9] P. Beran and W. Silva, *Reduced-order modeling: New approaches for computational physics*, in AIAA 2001-0853, 39th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, January 2001.

[10] J. Blazek, *Computational Fluid Dynamics: Principles and Applications*, Elsevier, first ed., 2001.

[11] M. Buffoni, H. Telib, and A. Iollo, *Iterative methods for model reduction by domain decomposition*, Computers & Fluids, 38 (2009), pp. 1160 – 1167.

[12] M. Cardoso, L. Durlofsky, and P. Sarma, *Development and application of reduced-order modeling procedures for subsurface flow simulation*, International Journal for Numerical Methods in Engineering, 77 (2009), pp. 1322–1350.

[13] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Scientific Computing, 32 (2010), pp. 2737–2764.

[14] E. Cheney and D. Kincaid, *Numerical Mathematics and Computing*, Thomson Brooks/Cole, Belmont, 2008.

[15] European Commission, *Flightpath 2050: Europe's Vision for Aviation*, report of the high level group on aviation research, European Union, 2011.

[16] T. Faber, *Fluid Mechanics for Physicists*, Cambridge University Press, Cambridge, 1995.

[17] M. Feistauer, J. Felcman, and I. Straskraba, *Mathematical and Computational Methods for Compressible Flow*, Clarendon Press, Oxford, 2003.

[18] J. Ferziger and M. Perić, *Computational Methods for Fluid Dynamics*, Springer, Berlin, third ed., 2002.

[19] A. Forrester, A. Sobester, and A. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*, Wiley, 2008.

[20] R. Fox, P. Pritchard, and A. McDonald, *Introduction to Fluid Mechanics*, Wiley, Hoboken, seventh ed., 2009.

[21] M. Galle, T. Gerhold, and J. Evans, *Parallel computation of turbulent flows around complex geometries on hybrid grids with the dlr-tau code*, in Proceedings of the 11th Parallel CFD Conference, A. Ecer and D. Emerson, eds., Williamsburg, VA, 23-26 May 1999, North Holland.

[22] T. Gerhold, O. Friedrich, J. Evans, and M. Galle, *Calculation of complex three-dimensional configurations employing the DLR-TAU-code*, AIAA paper, 167 (1997).

[23] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins Univ Pr, 3rd ed., 1996.

[24] D. Gratton and K. Willcox, *Reduced-Order, Trajectory Piecewise-Linear Models for Nonlinear Computational Fluid Dynamics*, in Proceedings of the 5th SMA Symposium, January 2004.

[25] D. Hänel, *Molekulare Gasdynamik: Einführung in die kinetische Theorie der Gase und Lattice-Boltzmann-Methoden*, Springer, 2004.

[26] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmertry*, Cambridge, New York, 1996.

[27] E. Houghton and P. Carpenter, *Aerodynamics for engineering students*, Butterworth-Heinemann, Oxford, fifth ed., 2003.

[28] Institute of Aerodynamics and Flow Technology, *Technical Documentation of the DLR TAU-Code Release 2009.1.0*, tech. rep., Deutsches Zentrum für Luft- und Raumfahrt e.V., 2009.

[29] E. Jones, T. Oliphant, P. Peterson, et al., *SciPy: Open source scientific tools for Python*, 2001–.

[30] I. Kalashnikova and M. Barone, *On the stability and convergence of a Galerkin reduced order model (ROM) of compressible flow with solid wall and far-field boundary treatment*, International Journal for Numerical Methods in Engineering, 83 (2010), pp. 1345–1375.

[31] G. Kerschen, J.-C. Golinval, A. Vakakis, and L. Bergman, *The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview*, Nonlinear Dynamics, 41 (2005), pp. 147–169.

[32] J. Klenner, K. Becker, M. Cross, and N. Kroll, *Future Simulation Concept*, in CEAS Conference Berlin, Session Numerical Simulation, Paper No. 1, 2007.

[33] K. Königsberger, *Analysis 2*, Springer, Berlin, fifth ed., 2004.

[34] P. Kundu and I. Cohen, *Fluid Mechanics*, Academic Press, San Diego, second ed., 2002.

[35] S. Lecheler, *Numerische Strömungsberechnung*, Vieweg+Teubner, Wiesbaden, 2009.

[36] P. LeGresley and J. Alonso, *Investigation of non-linear projection for pod based reduced order models for aerodynamics*, in AIAA Paper 2001-0926, 39th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, January 8-11 2001.

[37] ——, *Dynamic domain decomposition and error correction for reduced order models*, in AIAA Paper 2003-0250, 41st AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, January 6-9 2003.

[38] D. Luchtenburg, B. Günther, B. Noack, R. King, and G. Tadmor, *A generalized mean-field model of the natural and high-frequency actuated flow around a high-lift configuration*, Journal of Fluid Mechanics, 623 (2009), pp. 283–316.

[39] D. Lucia, P. Beran, and W. Silva, *Reduced-order modeling: new approaches for computational physics*, Progress in Aerospace Sciences, 40 (2004), pp. 51–117.

[40] D. Lucia, P. King, and P. Beran, *Reduced order modeling of a two-dimensional flow with moving shocks*, Computers & Fluids, 32 (2003), pp. 917 – 938.

[41] M. Mifsud and R. Zimmermann and J. Sippli and S. Görtz, *A POD-based reduced order modeling approach for the efficient computation of high-lift aerodynamics*, in Evolutionary and deterministic methods for design, optimization and control, C. Poloni, D. Quagliare, J. Périaux, N. Gauger, and K. Giannakoglou, eds., CIRA, Capua, Italy, 2011, EUROGEN 2011, pp. 213–225.

[42] K. Madsen, H. Nielsen, and O. Tingleff, *Methods for non-linear least squares problems (2nd ed.)*, 2004.

[43] G. Moore, *Cramming More Components onto Integrated Circuits*, Proceedings of the IEEE, 86 (1998), pp. 82–85.

[44] J. Moré, B. Garbow, and K. Hillstrom, *User Guide for MINPACK-1*, Tech. Rep. ANL-80-74, Argonne National Laboratory, Argonne, IL, USA, Aug. 1980.

[45] R. Pinnau, *Model reduction via proper orthogonal decomposition*, in Model Order Reduction: Theory, Research Aspects and Applications, Springer, 2008, pp. 95–109.

[46] M. Powell, *A hybrid method for nonlinear equations*, Numerical methods for nonlinear algebraic equations, 7 (1970), pp. 87–114.

[47] M. Rathinam and L. Petzold, *A new look at proper orthogonal decomposition*, SIAM Journal on Numerical Analysis, 41 (2004), pp. pp. 1893–1925.

[48] M. Rewienski and J. White, *A Trajectory Piecewise-Linear Approach to Model Order Reduction and Fast Simulation of Nonlinear Circuits and Micromachined Devices*, in Proc. Int. Conf. Computer-Aided Design, 2001, pp. 252–257.

[49] M. Rewienski and J. White, *A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices*, IEEE Trans. on CAD of Integrated Circuits and Systems, 22 (2003), pp. 155–170.

[50] F. Riesz and B. S. Nagy, *Vorlesungen über Funktionalanalysis*, VEB Deutscher Verlag der Wissenschaften, Berlin, 1956.

[51] C.-C. Rossow, *Einführung in die numerischen Methoden in der Aerodynamik.* Lecture notes, Wintersemester 2009/10.

[52] C.-C. Rossow and N. Kroll, *Numerical Simulation - Complementing Theory and Experiment as the Third Pillar in Aerodynamics*, in Hermann Schlichting–100 Years: Scientific Colloquium Celebrating the Anniversary of His Birthday, Braunschweig, Germany 2007, R. Radespiel, C.-C. Rossow, and B. W. Brinkmann, eds., Notes on Numerical Fluid Mechanics and Multidisciplinary Design, Springer, 2009.

[53] C. Rowley, T. Colonius, and R. Murray, *Model reduction for compressible flows using POD and Galerkin projection*, Physica D: Nonlinear Phenomena, 189 (2004), pp. 115–129.

[54] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 2nd ed., April 2003.

[55] M. Salas, *Digital Flight: The Last CFD Aeronautical Grand Challenge*, J. Sci. Comput., 28 (2006), pp. 479–505.

[56] R. Schaback and H. Werner, *Numerische Mathematik*, Springer, 4th ed., 1992.

[57] L. Sirovich, *Turbulence and the dynamics of coherent structures. Part I: Coherent structures*, Quarterly of applied mathematics, 45 (1987), pp. 561–571.

[58] G. Söderlind, *The logarithmic norm. history and modern theory*, BIT Numerical Mathematics, 46 (2006), pp. 631–652.

[59] D. C. Sorensen and S. Chaturantabut, *A State Space Error Estimate for POD-DEIM Nonlinear Model Reduction*, Siam Journal on Numerical Analysis, 50 (2012), pp. 46–63.

[60] G. Stewart, *On the early history of the singular value decompostion*, SIAM Review, 35 (1993), pp. 551–566.

[61] E. Tinoco, D. Bogue, T.-J. Kao, N. Yu, P. Li, and D. Ball, *Progress toward CFD for full flight envelope*, Aeronautical Journal, 109 (2005), pp. 451–460.

[62] D. Vasilyev, M. Rewienski, and J. White, *Macromodel Generation for BioMEMS Components Using a Stabilized Balanced Truncation Plus Trajectory Piecewise-Linear Approach*, IEEE Trans. on CAD of Integrated Circuits and Systems, 25 (2006), pp. 285–293.

[63] A. Vendl and H. Fassbender, *Efficient POD-based Model Order Reduction for steady aerodynamic applications*, in Evolutionary and deterministic methods for design, optimization and control, C. Poloni, D. Quagliare, J. Périaux, N. Gauger, and K. Giannakoglou, eds., CIRA, Capua, Italy, 2011, EUROGEN 2011, pp. 296–309.

[64] A. Vendl and H. Fassbender, *Missing point estimation for steady aerodynamic applications*, PAMM, 11 (2011), pp. 839–840.

[65] ——, *Projection-Based Model Order Reduction for Steady Aerodynamics*, in Notes on Numerical Fluid Mechanics and Multidisciplinary Design, N. Kroll, R. Radespiel, J. van der Burg, and K. Sorensen, eds., vol. 123, Springer, Heidelberg, to appear in 2013.

[66] V. Venkatakrishnan, *Implicit schemes and parallel computing in unstructured grid CFD*, ICASE Report No. 95-28, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, 1995.

[67] S. Volkwein, *Model reduction using proper orthogonal decomposition.* Lecture Notes, Institute of Mathematics and Scientific Computing, University of Graz. URL: http://www.uni-graz.at/imawww/volkwein/POD.pdf.

[68] T. Voss, A. Verhoeven, T. Bechtold, and J. ter Maten, *Model order reduction for nonlinear differential algebraic equations in circuit simulation*, in Progress in Industrial Mathematics at ECMI 2006, Series Mathematics in Industry 12, Springer, 2007, pp. 518–523.

[69] F. White, *Fluid Mechanics*, McGraw Hill, fourth ed., 1999.

[70] D. Whitfield and J. Janus, *Three-Dimensional Unsteady Euler Equations Using Flux Vector Splitting*, AIAA Journal, 84-1552 (1984).

[71] K. Willcox, *Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition*, Computers & Fluids, 35 (2006), pp. 208 – 226.

[72] R. Zimmermann, *Towards best-practice guidelines for POD-based reduced order modeling of transonic flows*, in Evolutionary and deterministic methods for design, optimization and

control, C. Poloni, D. Quagliare, J. Périaux, N. Gauger, and K. Giannakoglou, eds., CIRA, Capua, Italy, 2011, EUROGEN 2011, pp. 326–341.

[73] R. Zimmermann and S. Görtz, *Improved Extrapolation of Steady Turbulent Aerodynamics using a Non-Linear POD-Based Reduced Order Model*, The Aeronautical Journal, Preprint (2012).

[74] R. Zimmermann, S. Görtz, M. Mifsud, and A. Vendl, *Surrogate Modeling for AeRo data Toolbox (SMART) - POD Tool Technical Documentation*, tech. rep., $C^2A^2S^2E$, DLR, 2011.

Index